1994年2月1日発行（毎月1回1日発行）第13巻2号通巻142号　昭和58年11月2日第三種郵便物認可

PERSONAL COMPUTER MAGAZINE for MZ, X1, and X68000

Oh!X

特集 X-BASICとグラフィック

マンデルブロ集合を描画する/降雪シミュレータ/スプライトを扱う

1993年度GAME OF THE YEARノミネート発表

連載 ワンチップIC工作入門/ハイパーピクセルワークス

2
1994



SOFT BANK
オー／エックス
定価600円


SHARP
目の付けどころが、
シャープでしょ。
夢の、頂きへ。
68ワールドの最高峰。
X68030
32bit PERSONAL WORKSTATION

32ビットパーソナルワークステーション

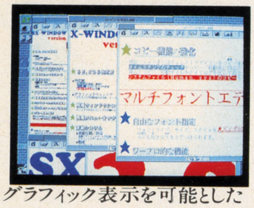
演算速度4.3倍(当社10MHz機比)/2.4倍(当社XVI比)※1、動画ウィンドウに見る新創造次元。選ばれた人だけが持つ感性によってX68030の扉はひらかれる。

X68000シリーズとして初の32ビットMPU MC68EC030を搭載して高速化を実現。

データキャッシュ、プログラムキャッシュをそれぞれ256バイト搭載したクロック周波数25MHzの高速32ビットMPUを搭載。演算速度は2倍以上上(当社従来比)※1の高速化を実現しました。また数値演算プロセッサMC68882※2(25MHz)もサポート。大量の実数演算を必要とするクリエイティブワークやGUI環境の操作性など、実行速度の飛躍的な向上が図られています。(当社従来比)

※1 Dhrystn(四則演算)比。25MHz・データキャッシュオン・プログラムキャッシュオンでMC68000/10MHz時の約4.3倍、16MHz時の約2.4倍。
※2 数値演算プロセッサCZ-5MP1標準価格54,800円(税別):本体内の専用ソケットに取りつけ可能。

65,536色表示、動画表示を実現。さらにパワーアップしたSX-WINDOWver.3.0。



X68000独自のウィンドウシステムとして定評の「SX-WINDOWver.2.0」をさらに強化した「SX-WINDOWver.3.0」を標準装備。新たに、65,536色の自然色グラフィック表示を可能とした『グラフィックウィンドウ』※を搭載。またアニメーション動画をウィンドウ上で表現でき、手軽にコンピュータアニメーションが楽しめる『CGAウィンドウ』、さらに従来のエディタのイメージを一新、高度な日本語文書作成をサポートするSX-WINDOW対応の高機能日本語マルチフォントエディタを標準装備。アウトラインフォントの展開もさらに高速化が図られています。

※SX-WINDOW上の512×512ドットのエリア内で表示可能。

GUIに対応する大容量メインメモリを搭載。

メインメモリは標準で4Mバイト、複数のアプリケーションをウィンドウ上で同時に使用するなど大量のデータ処理に対応。また本体内の増設で、I/Oスロットを使用せず最大12Mバイトまで拡張できます。拡張したメモリはすべて32ビットバスによる高速アクセスが可能、優れた拡張環境でシステムパワーアップをサポートします。

※メモリ増設には、4MB内部増設RAMボードCZ-5BE4標準価格54,800円(税別)、4MB増設RAMモジュールCZ-5ME4標準価格49,800円(税別)をご使用ください。なおCZ-5ME4はCZ-5BE4上に装着します。

X68000シリーズの高機能を継承した上で、さらに使いやすさの向上を図ったコンパチビリティ重視設計※1、すぐに使える高機能ソフトを標準装備。

●25MHzでは速すぎるアプリケーションも、従来のクロック周波数(10MHz/16MHz)で動作可能なソフトコンパチ重視設計●65,536色同時発色の自然色グラフィックス(最大表示エリア512×512ドット)、1024×1024ドットの実画面エリアを持つ高解像度表示能力(最大表示エリア768×512ドット)、疑似高解像度スーパーインポーズ(インターレース方式/512×480ドット・専用ディスプレイテレビ使用時)を装備した高精細度自然色グラフィックス機能。●外部MIDI音源もコントロール可能※2、ウィンドウ上で手軽にコンピュータミュージックが楽しめるMIDI音源対応デバイスドライバ搭載●ステレオ8オクターブ8重和音FM音源、ADPCM搭載●プリンタ、RS-232C、SCSI、オーディオ入出力、イメージ入力など多彩なインターフェイスを装備。●日本語変換効率や操作性を高めた日本語フロントプロセッサASK68Kver3.0搭載。●従来のエディタのイメージを一新したSX-WINDOW対応の高速多機能日本語マルチフォントエディタ標準装備●日本語マルチフォントエディタ中に貼り付ける絵やグラフなどが簡単に作成できるグラフィックパターンエディタ●MIDI対応のX-BASIC。

※1 アプリケーションソフトおよび周辺機器のうち、一部動作しないものがあります。詳しくはシャープお客様相談窓口にお問い合わせください。
※2 別売のMIDIインターフェイスが必要です。

130mmFD(5.25型) マンハッタンシェイプシリーズ



■X68000伝統のマンハッタンシェイプを継承 ■130㎜FDD(5.25型)2基搭載 ■80MBハードディスク内蔵(CZ-510C)※ ■マウス・トラックボール標準装備 ■ASCII準拠フルキーボード採用

※CZ-500Cには、80MB内蔵用ハードディスクドライブCZ-5H08/160MB内蔵用ハードディスクドライブCZ-5H16を用意しています。

X68030
32bit PERSONAL WORKSTATION

本体+キーボード+マウス・トラックボール
130㎜(5.25型)FDタイプ
CZ-500C-B(チタンブラック)標準価格398,000円(税別)
HD内蔵 CZ-510C-B(チタンブラック)標準価格488,000円(税別)
14型カラーディスプレイ
CZ-608D-B(チタンブラック)標準価格94,800円(税別・チルトスタンド同梱)

90mmFD(3.5型) コンパクトシリーズ

■32ビットのハイパワーを凝縮したコンパクトフォルム ■2DD対応90㎜FDD(3.5型)2基搭載 ■80MBハードディスク内蔵(CZ-310C)※ ■マウス標準装備 ■コンパクトキーボード採用

※CZ-300Cには、80MB内蔵用ハードディスクドライブCZ-5H08/160MB内蔵用ハードディスクドライブCZ-5H16を用意しています。

X68030
32bit PERSONAL WORKSTATION
Compact

本体+キーボード+マウス
90㎜(3.5型)FDタイプ
CZ-300C-B(チタンブラック)標準価格388,000円(税別)
HD内蔵 CZ-310C-B(チタンブラック)標準価格478,000円(税別)
14型カラーディスプレイ
CZ-608D-B(チタンブラック)標準価格94,800円(税別・チルトスタンド同梱)



X68030
32bit PERSONAL WORKSTATION
&
X68000
PERSONAL WORKSTATION ·XVI

68買ったら EXEクラブへ 入ろう!

EXE クラブって何だ?

X68030/X68000を手に入れたら、やっぱり他のユーザーがどんな風に使っているのか気になるもの。ということでEXEクラブは、そんなあなたのための、他の68ユーザーとのコミュニケーションをバックアップする、情報交換の場です。

本体同梱の入会申込ハガキを送るだけで、自動的に無料入会。さらに下記の特典付き。

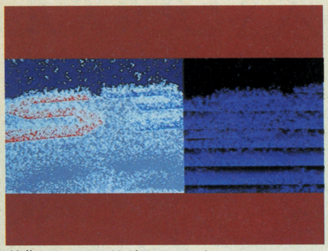
メリット1 会員ナンバー入りオリジナル会員電卓がもらえる。

メリット2 各種フェアご優待・イベント案内等、数々の特典がある。

●消費税及び配送・設置・付帯工事費、使用済み商品の引き取り費等は、標準価格には含まれておりません。

■お問い合わせは… シャープ株式会社 コンシューマーセンター 西日本相談室 〒545 大阪市阿倍野区長池町22番22号 ☎(06)621-1221(大代表)
電子機器事業本部システム機器営業部 〒545 大阪市阿倍野区長池町22番22号 ☎(06)621-1221(大代表)

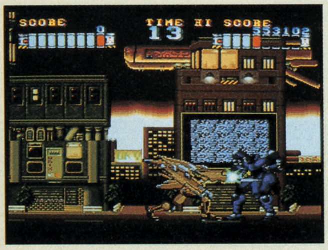
資料請求券 X68030 Oh!X 2係



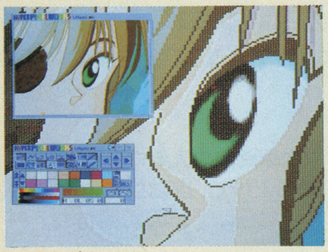
特集　X-BASICとグラフィック



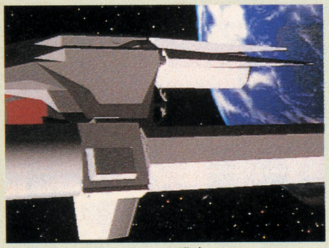
キーパー



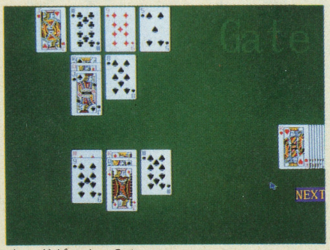
マッドストーカー



ハイパーピクセルワークス



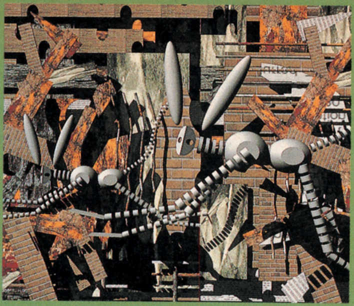
DōGA CG アニメーション講座



カードゲーム　Gate

CONT

〈スタッフ〉
●編集長／前田　徹　●副編集長／植木章夫　●編集／山田純二　豊浦史子　高橋恒行　●協力／有田隆也　中森　章　林　一樹　吉田幸一　華門真人　吉田賢司　朝倉祐二　大和　哲　村田敏幸　丹　明彦　三沢和彦　長沢淳博　司馬　護　清瀬栄介　石上達也　柴田　淳　瀧　康史　横内威至　進藤慶到　●カメラ／杉山和美　●イラスト／山田晴久　江口響子　高橋哲史　川原由唯　●アートディレクター／島村勝頼　●レイアウト／元木昌子　ADGREEN　●校正／グループごじら



表紙絵：塚田　哲也

1994 FEB. 2

E　N　T　S

■広告目次

先が、面白くなる。

ウィンドウ環境のプラットホームを確立、SX-WINDOW ver.3.0



■実画面：1,024×1,024ドット、表示画面：768×512ドット

●この画面は広告用に作成した、機能を説明するためのイメージ画面です。また、各種アイコン等は、SX-WINDOW ver.3.0がもつ機能を使って作成したもので、標準装備のものとは異なるものもあります。
●本広告中のエディタで表示している文字のフォントはZeit社の、『書体倶楽部』のフォントを使用しています。

シャープ株式会社　●お問い合わせは…コンシューマーセンター西日本相談室　〒545 大阪市阿倍野区長池町22番22号 ☎(06)621-1221(大代表)
電子機器事業本部システム機器営業部 〒545 大阪市阿倍野区長池町22番22号 ☎(06)621-1221(大代表)

資料請求券
CZソフト
Oh!X
2係

SHARP

目の付けどころが、シャープでしょ。

に見たGUIの新展開。

❶マルチフォントエディタ編集例。文字ごとに文字種、文字の大きさの指定、修飾が可能で、イメージデータの貼り付けもOK。

❷CONFIG.SYSやAUTOEXEC.BATなどの編集に便利な「エディタ」モードの例。このように日本語マルチフォントエディタは、用途に合わせてカスタマイズできます。

❸❶の画面をプリンタで印字した例。対応プリンタも増えました。(カラー印刷は誤差分散により65,536色対応)

❹「パターンエディタ」で作成したデータを、背景に設定できます。

❺バージョンアップした日本語フロントプロセッサASK68K ver.3.0の辞書メンテナンスがウィンドウ上で可能。

❻アイコンデータや背景データを作成する「パターンエディタ」。文字の貼り付けなど、編集機能も一段とフレンドリーに。

❼オリジナルに作成したアイコンパターンの例。

❽512×512ドットの範囲内で65,536色の表示が可能。

❾さまざまなグラフィックフォーマットに対応しています。

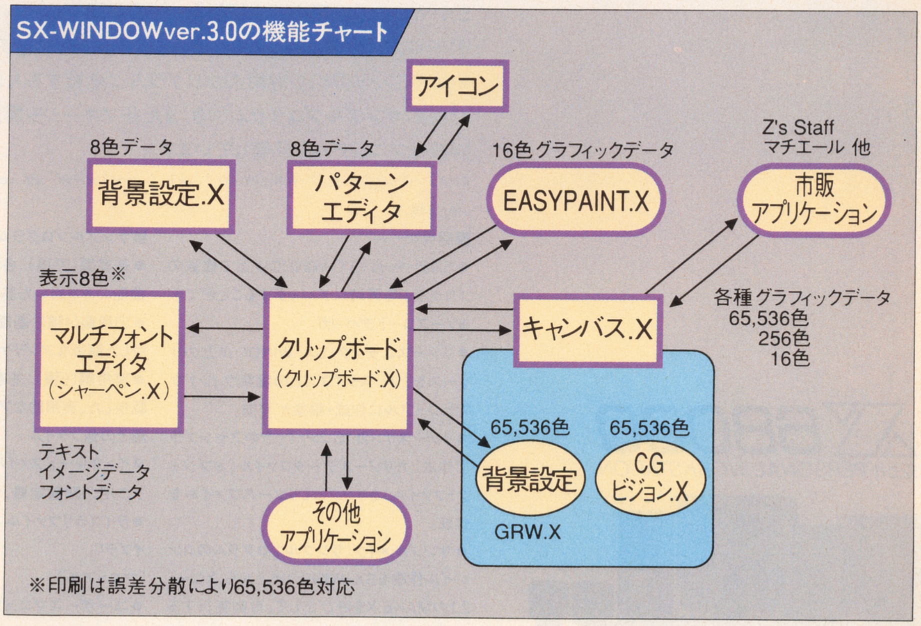
❿任意のサイズに縮小・拡大表示可能。

⓫異なる画像フォーマットへのコンバートができます。

⓬「CGAウィンドウ」、65,536色(最大)のコンピュータアニメーション表示が可能です。

発展性のあるプラットホームとしてのウィンドウシステム、
SX-WINDOW ver.3.0が提供する新たなGUI環境が
さらなるウィンドウ時代を予見する――――。
国産オリジナルウィンドウとしての意味、未来への確かなビジョン、
ユーザーインターフェイスや高速化へのゆるぎない探求が
ここに凝縮されています。
65,536色表示はもちろん、さまざまな画像フォーマット対応、
イメージデータのコピー&ペースト、
動画、音楽/音声再生をサポートするマルチメディア環境。
そして、何よりもこれらが密接に連携して
統合的にハンドリングできるエキサイティングな環境を創造しています。
未来を照準に入れたウィンドウアーキテクチャ、
そのインテリジェンスがいよいよX68030/X68000シリーズで享受できます。

SX-WINDOWver.3.0の機能チャート



X68030
32bit PERSONAL WORKSTATION

X68000
PERSONAL WORKSTATION・XVI

X68030　X68030 Compact　X68000 XVI　X68000 XVI Compact



SHARP

目の付けどころが、シャープでしょ。

For X68030/X68000series APPLICATION SOFTWARE

◎定評のあるGUI対応のウィンドウワープロ。

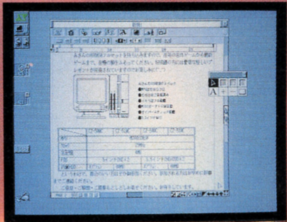
EGWord SX-68K

CZ-271BWD 2月発売予定

ウィンドウワープロとして評価の高いEGWordのSX-WINDOW対応版。キャラクタベースのワープロを超えたグラフィカル・ユーザーインターフェイス(GUI)による手軽なDTPソフトとしても優れた表現力を発揮します。定評ある日本語入力方式(EGConvert)によるインライン入力、文書互換を実現するEDF形式もサポートしています。

NEW





●禁則処理を生かした美しく、読みやすい文書作成：文字間隔を自然に美しく配置。さらに均等禁則など、豊富な禁則処理によるきめ細かな調整が可能。

●作図感覚で罫線・作表作業も快適：マウスによる作図感覚の作表、また豊富な線種で、行や列を気にせずに文字と文字の間に罫線が引けます。表組も自在に編集できるとともに、罫線で囲まれたブロック単位で網掛けも可能。

●DTPに迫る多段組、レイアウト表示：段組は2段から5段まで設定でき、段間隔の調整や段組線の表示はもちろん、自由な位置での改段も可能。レイアウト表示もOK。

●様々なグラフィックデータやテキストデータの貼り込みが可能：他のソフトで作成された色々な画像データ(GScript形式)をEGWordの文書に取り込めます。もちろん取り込んだ画像の編集もOK。

●短文・書式登録でルーチンワークの負担を軽減●充実した国語辞書機能に優れた専門辞書をプラス●実用性の高い逐次自動変換方式を採用●ウィンドウの特性を生かした優れたユーザーインターフェイス●ルーラによるスピーディ&イージーな書式設定●文書互換を実現するEDF(Extended Document Format)形式をサポート。

*5MB以上の空きのあるハードディスクが必要です。 4MB、ver.2.0

◎待望のSX-WINDOW開発支援ツール。

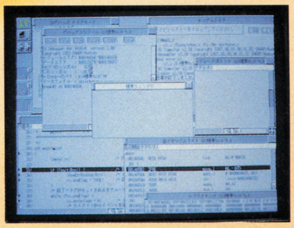
SX-WINDOW 開発キット Workroom SX-68K

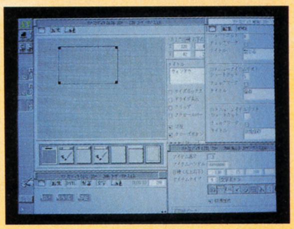
CZ-288LWD 2月発売予定

SX-WINDOW用のソフト開発に必要な開発ツールやサンプルプログラムを装備。プログラムの編集、リソースの作成、コンパイル、デバッグといった一連の作業をSX-WINDOW上で効率よく実行できます。初めてSX-WINDOW用のプログラムに挑戦する人にも、簡単に基本機能の理解ができる33種のサンプルプログラム付き。また各マネージャ解説と関数リファレンスの詳細なマニュアルも装備しています。

NEW

※メインメモリ4MB以上、SX-WINDOW ver.2.0以上、C compiler PRO-68K ver.2.1が必要です。





〈キット構成〉

■開発ツール

●SXデバッガ：SX-WINDOW上で複数のプログラムを同時にデバッグすることができるソースコードデバッガ。

●リソースエディタ：SX-WINDOW上のリソースをリソースタイプごとの編集ウィンドウでビジュアルに作成・編集が可能。

●リソースリンカ：Cコンパイラやアセンブラで作成したリソースデータファイル(オブジェクトファイル)をリンクしてリソースファイルを作成。

●サンプルメイク：サンプルプログラムのコンパイル作業をSX-WINDOW上から、XCver2.1のMAKE.Xを呼び出して、自動実行する簡易メイクユーティリティ。

■サンプルプログラム

●基礎編(23種)：各マネージャの基本的な機能のみを用いた基本動作の理解。

●応用編(4種)：基礎編での基本機能を応用した簡単なアプリケーションの作成。

●実用編(6種)：基礎/応用編での機能を駆使した、実用的なアプリケーションの作成。

■その他ファイル

●インクルードファイル：Cコンパイラとアセンブラ用の関数定義、データ定義ファイル。

●ライブラリファイル：Cコンパイラ用関数ライブラリ。

[マニュアル]

●ユーザーズマニュアル●プログラマーズマニュアル●SXライブラリマニュアル

X68030
32bit PERSONAL WORKSTATION



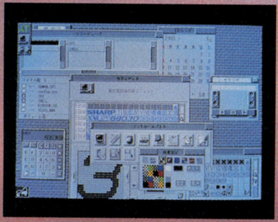
その先のシーンへ。

●65,536色対応、動画ウィンドウ標準装備。

SX-WINDOW ver3.0 システムキット

CZ-294SS(130㎜FD)/CZ-294SSC(90㎜FD)各標準価格19,800円(税別)

自然描画に迫る美しい表現が可能な65,536色表示のグラフィックウィンドウを装備。さらにグラフィックウィンドウ内でのアニメーション動画表示、各種グラフィックデータのコンバートも実現しました。またイメージデータの貼り付けなどをサポートした日本語マルチフォントエディタを始め、クリエイティブワークを支援する数々の便利機能を装備、Human68k ver.3.0システムディスクも付属しています。



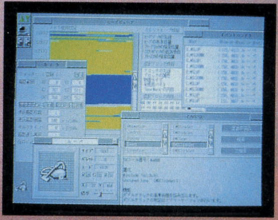
※メインメモリ4MB以上必要です。SX-WINDOW ver.1.0/1.1/2.0をお持ちの方には有償バージョンアップを行っています。

●「SX-WINDOW開発キット」のサポートツール。

開発キット用ツール集

CZ-289TWD　2月発売予定　NEW

SX-WINDOW開発キットをさらに使いやすくするためのツールです。SXコールの簡易リファレンスを簡単に検索するインサイドSX、イベントの発生を常時監視確認するイベントハンドラ、リアルタイムにメモリブロックの利用状況を表示するヒープビューアなど11種のツールが用意されています。(2MB、ver.2.0)



●SX-WINDOW対応ドローイングツール。

Easydraw SX-68K

CZ-264GWD　標準価格19,800円(税別)

ホビーからビジネスまで幅広い分野で活用できる、待望のドローイングツールです。イラスト、フローチャート、地図、見取り図など各種グラフィックが製図感覚で作成できます。作成したデータは、他のSX-WINDOW対応アプリケーションでも利用でき、企画書やプレゼンテーション資料の作成をサポートします。またレーザープリンタドライバを付属、このドライバはSX-WINDOW対応の他のアプリケーションでも利用することができます。(4MB、ver.3.0)



●SX-WINDOWを楽しく使うためのアクセサリ集。

SX-WINDOW デスクアクセサリ集

CZ-290TWD　標準価格14,800円(税別)

SX-WINDOWをさらに便利に、楽しく使うためのデスクアクセサリ集です。スクリーンセーバ、アドレス帳、電子手帳通信ツール、パズルなど12種類の豊富なアクセサリが収められています。

1キーノート2スクリーンセーバ3スクラップブック4ミュージックボックス5ハイパーリンク(電子手帳通信ツール)6アドレス7スケジューラ8ウィンドウアイコニファイ9ソフトウェアキーボード10パズル11ファイルサーチ(ファイル検索ツール)12フォントリンカ。(2MB、ver.3.0)



●マルチタスク機能をはじめ、通信環境がさらに充実。

Communication SX-68K

CZ-272CWD　標準価格19,800円(税別)

通信環境をさらに高めたウィンドウ対応の通信ソフトです。マルチタスク機能により他のアプリケーションソフトを実行中でも簡単に通信が可能。また、ホスト局をクリックするだけの自動ログイン機能、初心者にも簡単なプログラム機能、最新モデム(20種類)もフルサポートしています。(2MB、ver.1.1)

●ウィンドウ対応グラフィックツール。

Easypaint SX-68K

CZ-263GWD　標準価格12,800円(税別)

マウスによる簡単操作、65,536色中16色の多彩な表現、クリエイティブマインドに応えるウィンドウ対応ペイントツールです。同時に複数のウィンドウを開いて編集でき、各ウィンドウ間でのデータ交換もできます。(2MB、ver.1.1)

●FM音源サウンドエディタ。

SOUND SX-68K

CZ-275MWD　標準価格15,800円(税別)

他のミュージックソフトで演奏中の音色を、簡単に作成、変更できるマルチタスク機能、またエディット、イメージ、ウェーブの3つの編集/確認モードを装備。作成中の音色も50曲の自動演奏でリアルタイムに確認、編集できます。まさにミキサー感覚で音創りが楽しめるツールです。(2MB、ver.1.1)

●SX-WINDOW対応になってさらにパワーアップ。

倉庫番リベンジ SX-68K ユーザー逆襲編

CZ-293AW(130㎜FD)/CZ-293AWC(90㎜FD)各標準価格6,800円(税別)

倉庫番10年にわたるユーザーの投稿など、新作306面が目白押し。まさに倉庫番の最強版がSX-WINDOW上で楽しめます。AI機能やエディット機能、キャラクタ変更機能も装備。半年で解けたらあなたは天才?です。(2MB、ver.1.1)

PRO-68K シリーズ

●X68030/X68000対応　NEW

C COMPILER PRO-68K ver.2.1 NEW KIT

CZ-295LSD　標準価格44,800円(税別)

※メインメモリ2MB以上が必要です。

C compiler PRO-68KのX68030/X68000対応版。MPU68030、MC68882の命令セットに対応したアセンブラ、デバッガ、ソースコードデバッガを付属。またHuman68k ver.3.0、ASK68K ver.3.0にも対応。新たにGPIBライブラリ、MC68882対応フロートライブラリを付属しています。



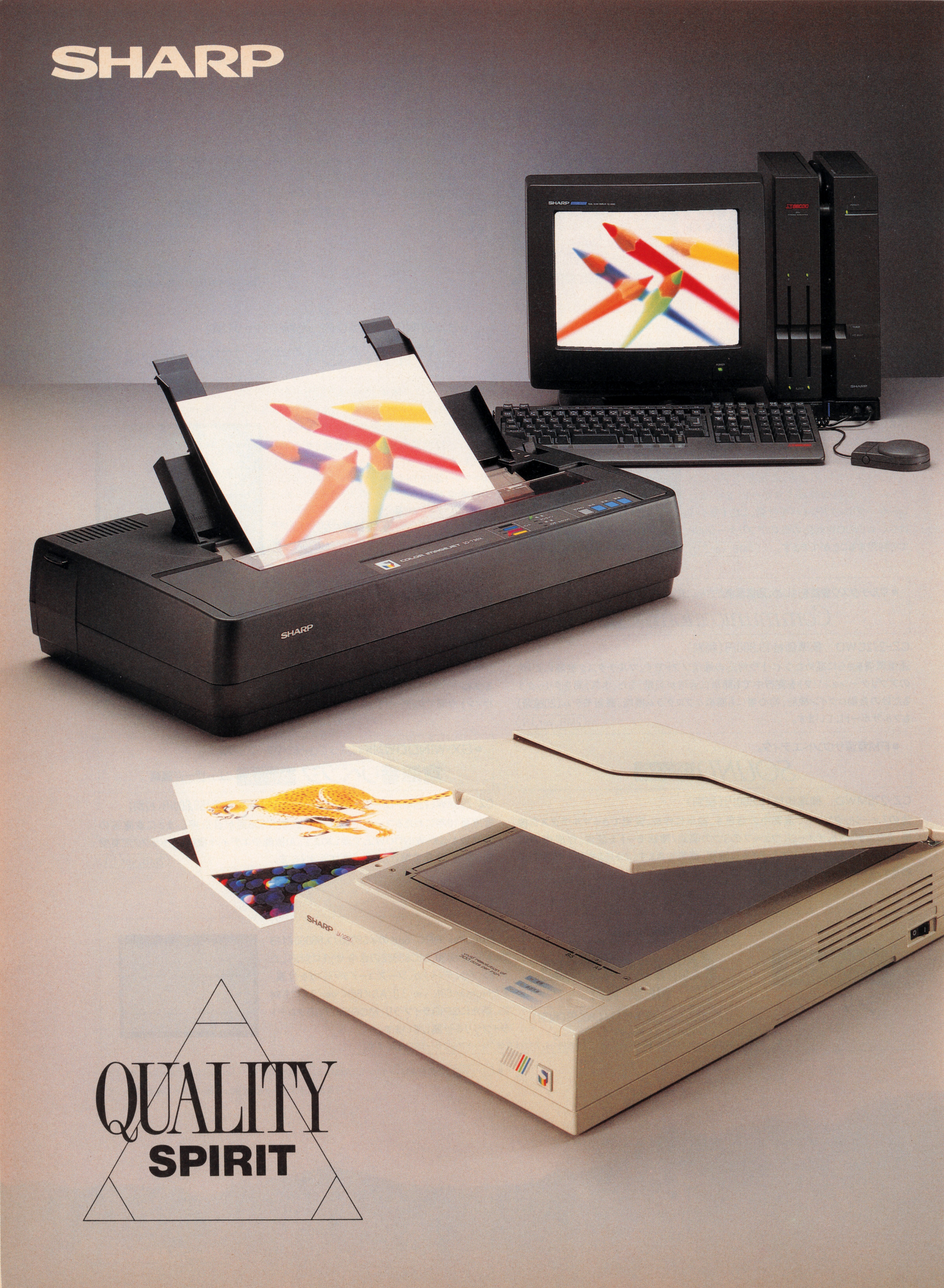
※C compiler PRO-68K/ver.2.0/ver.2.1をお持ちの方には有償グレードアップサービスを行います。

※(2MB,ver.1.1)の表示は、メインメモリ2MB以上、SX-WINDOW ver.1.1以上が必要であることを示します。

※発売予定のソフトの画面は実物とは異なる場合があります。

●お問い合わせは…シャープ㈱電子機器事業本部(液映)システム機器推進プロジェクトチーム　〒162　東京都新宿区市谷八幡町8番地　☎(03)3260-1161(大代表)へ　シャープ株式会社

資料請求券　CZソフト　Oh!X　2/係


SHARP
SHARP
SHARP
QUALITY
SPIRIT

ツクモパソコン本店II 3F、OAシステムプラザ（札幌・仙台・横浜西口・
大須・大阪日本橋・岡山・広島・福岡・鹿児島）でも販売中／

宍戸留美

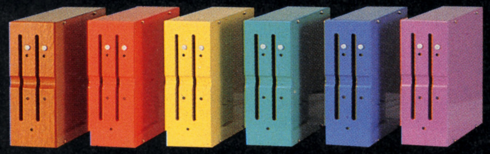
夢は、い・た・だ・きよ…。



コンパクト XVI 改造機。
弊社にて1年保証。クロックは10／16／24の3モード。16／24MHzは背面トグルスイッチにより切替。RED ZONEの24MHz では正常動作しないソフト等がありますが、10／16MHz でご使用になれます。

X68000 Compact XVI 改

RED ZONE ¥160,000



・シャープ製CZ－6FD5完全コンパチブル・オートイジェクト機能付・ドライブ番号切替スイッチ付・木製（ナラ材）フロントパネル・対応機種／CZ－674C／300C／310C／500C／510C
・カラーリングオプションはプラス5,000円です。

①②③④⑤⑥

シャープ製CZ-6FD5
完全コンパチFDD（MK-FD1）

満開式軟盤駆動装置壱號

¥39,800（税別、カラーモデル¥44,800）

Compact XVI 改　5インチFDD
RED ZONE ＋ MK－FD1
セット価格 **¥180,000**（税別）
（FDカラーモデルは＋¥5,000）

新登場 満開式硬盤駆動装置弐號
1GバイトSCSI ハードディスクユニット
MK-HD2
¥150,000→**¥125,000**（税別）
平均アクセスタイム8.3msの高速HDがこの価格
直販のみです。

98バスマウスを68で使っちゃうアダプタ
MOUSEJACK68-98
上記パソコンショップでもお求めになれます。
MK-MJ1 ¥4,000（税別）

当ショップは通販専門店です。X680×0用各種ハード・ソフトも取り扱っております。お電話にて商品リストと注文書をご請求ください。RED ZONEのご購入には承諾書が必要です。合わせてご請求ください。

〒171 東京都豊島区長崎1－28－23Muse西池袋2F
TEL (03)3554－7441　FAX (03)3554－3856

パソコンショップ満開
㈱満開製作所



すべては進化します。いまのSNKをごらんください。

アミューズメントの発想を進めるとすべての文化へ

人間の進歩の歴史はさまざまな分野で「壁」を破りつづけ、限りない可能性に挑む。そのくりかえしだったに違いありません。努力がまた新しい壁をつくり、人間はそれをみずからの英知でのりこえる。こんにちあるすべてのものは、この努力の積み重ねが生んだ結果といえるでしょう。私たちが標榜しているのは、あらゆる最先端のテクノロジーに遊びの発想をプラスし映像の世界をいかに楽しく、文化的に提案していくか。単なるゲームだけにとらわれず、技術に遊びと生活、文化の楽しみをプラスして全く違う可能性を切り開くことを意味しています。SNKはたえず夢と感動を提供しつづける使命があると考えます。そして、スーパーマルチメディア時代に向け三次元空間を自在に走りつづけます。だからいまのSNKをごらんください。
いつも新しい胎動が息づいています。

NEO GEO
The Future Is Now
SNK

1/60秒で走れるプログラマー募集。

■会社プロフィール

[設　　立]　昭和53年7月
[資 本 金]　8,800万円
[社 員 数]　760名(男580名、女180名)
[年　　商]　433億円(平成5年9月実績)
[関連会社]　SNK CORPORATION OF AMERICA
SNK HOME ENTERTAINMENT, Inc.
SNK ASIA LTD.
[事 業 所]　大阪本社、支店及び営業所: 全国45ヶ所

■事業内容

業務用TVゲーム、家庭用TVゲームのハード＆ソフトの企画・開発・製造・販売・リース、輸出、店舗運営

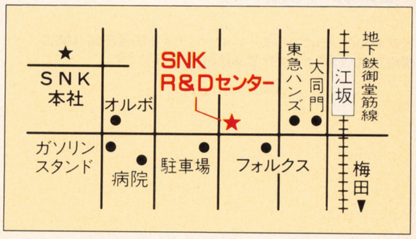
■募集要項

[職　　種]　プログラマー
[仕事内容]　ゲームソフトのプログラミング
[資　　格]
- X68000でアセンブラ言語によるゲームプログラミングの経験者
- 高卒以上20才～28才

[給　　与]　下記の金額は、あくまでも最低金額です。経験・年齢・能力・前給を充分に考慮の上、決定させていただきます。
固定給制　164,000円以上
(時間外手当有)
[待　　遇]　昇給年1回(93年実績6%)、賞与年2回、手当/家族・役職・休日出勤・時間外・交通費全額支給
[福利厚生]　各種社会保険完備、退職金制度、社員持株会、各種レクリエーション有
[休日休暇]　日曜日、土曜日、祝日、有給・慶弔・夏期・年末年始休暇有、年間休日121日
[勤 務 地]　SNK R＆Dセンター(江坂)
[時　　間]　9:00～17:30

■応　　募

履歴書(写真貼付)・職務経歴書(B5サイズ)を本社総務部採用課まで郵送して下さい。



株式会社エス・エヌ・ケイ 総務部 採用課
本社　〒564 大阪府吹田市豊津町18-8
☎:0120-21-6522 FAX:06(338)7150

microware®

業界初!!
パソコンの
新・使い方

パソコン上で
ビデオCDを見る。

X680×0上でフルモーション画像をラクラク再生。
ほら、びっくり、この通り！



▲画像提供：エイリアスジャパン社

「ビデオPC for X680×0」は、お手持ちのOS-9/X680×0に組み込むだけで、ビデオCDやカラオケCDを簡単に再生。X680×0の利用範囲がまた拡がりました。

＊ビデオCDは、CDにフルモーション・ビデオを画像圧縮（MPEG規格）して収録した、世界標準規格のメディアです。

■パッケージ内容
- ●MPEG A/Vデコーダボード1枚
- ●MPEG動画ソフト一式
- ●マニュアル

■ソフトウェア・サポート
- ●MPFMドライバ
- ●ビデオCDプレイ・ユーティリティ
- ●CD-ROMドライバ

＊適応CD-ROMドライブの機種については、基本的にCD-ROM XA対応で倍速以上のドライブユニット

■必要条件
- ●このパッケージをご利用の際には、別売りの『OS-9/X68000 V2.4』、または『OS-9/X68030 V2.4.5』が必要です。
- ●CD-ROMドライブは別途ご用意ください。

ビデオPC
for X680×0

¥58,000（税別）

フリーダイヤル0120-355209
マイクロウェア・システムズ株式会社
〒101 東京都千代田区外神田2-17-3
TEL.03-3257-9000㈹ FAX.03-3257-9200

Z-MUSIC システム Ver.2.0

ついにMUSICシステムの正式バージョンアップ版が登場します。
X68000の音源ドライバとしてさらに使いやすく高機能なものになりました。

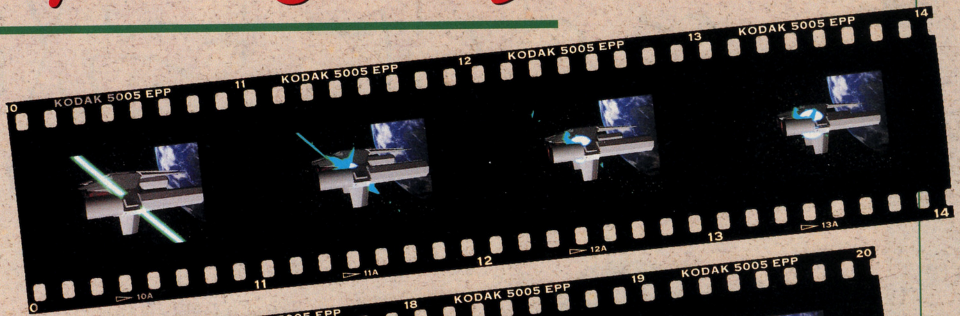
ver.1.0/1.1からのバージョンアップ内容

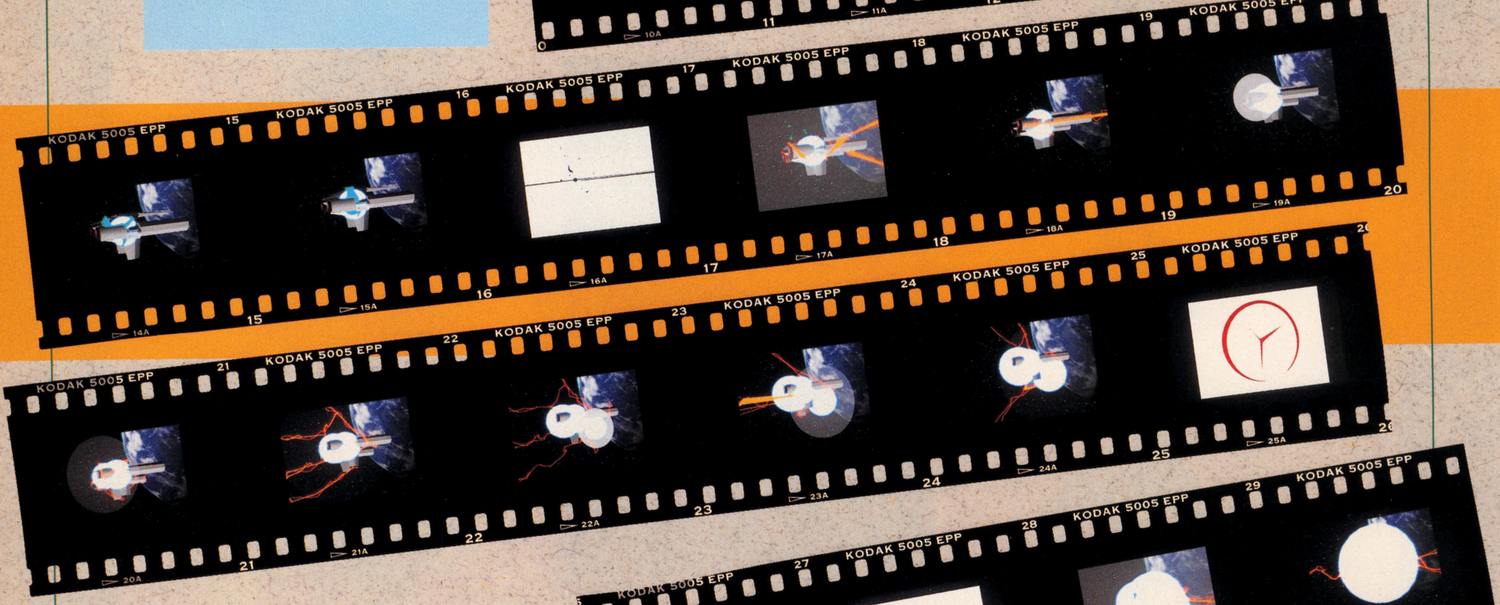
- PCM8対応AD PCM同時発音8声音量可変
- モジュレーション用波形メモリ搭載
- PCMバンクに対応
- ステップエディット系コマンド追加
- X68030完全対応ユニバーサルバージョン
- RS-232C対応版収録
- POLYPHON対応版収録
- 再生専用機能縮小版収録
- Cコンパイラ用ライブラリ完成
- AD PCM加工機能強化
- さらにクオリティを高めたAD PCMデータ
- もちろん、全ソースプログラム付属&ライセンスフリー

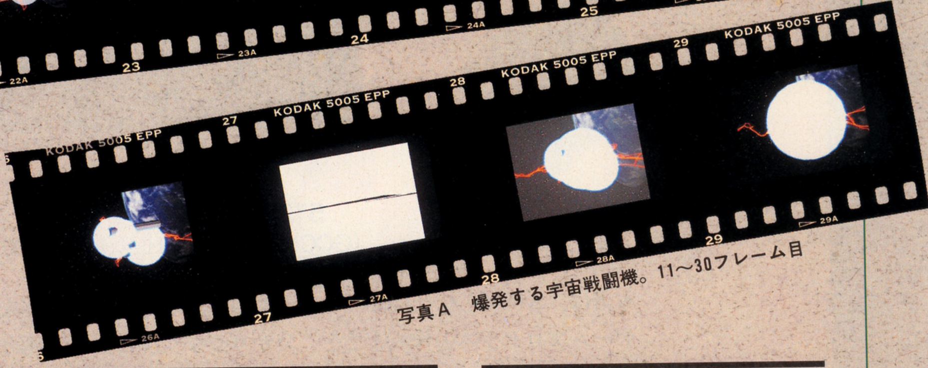


SOFT BANK ソフトバンク株式会社／出版事業部
〒103 東京都中央区日本橋浜町3-42-3 TEL03-5642-8100

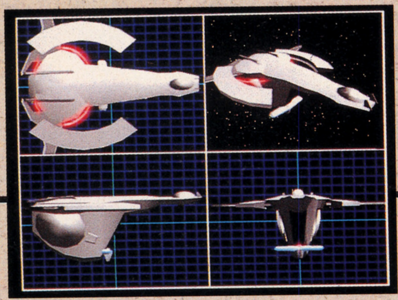
先月の続きの宇宙戦闘機の爆発を作成してみました。いかがでしょうか。このテクニックは，爆発だけではなく工夫次第でいろいろな演出に使うことができます。アニエフェの与え方の応用例もいくつか紹介しましょう。







写真A　爆発する宇宙戦闘機。11〜30フレーム目

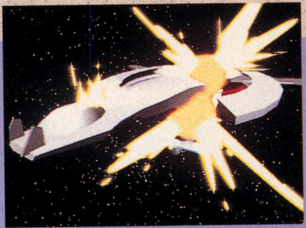


●今月のやられメカ

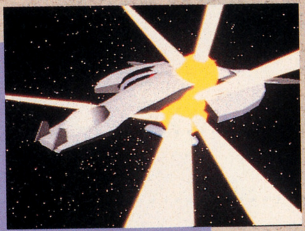
アニエフェの応用例



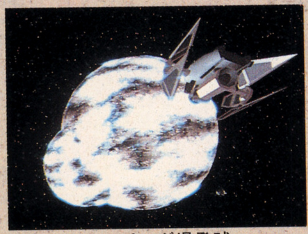
写真C　ギザギザ爆発



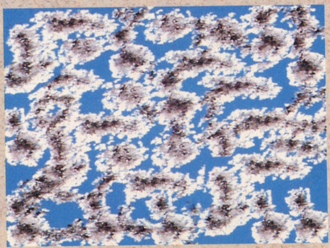
写真D　2色使用



写真E　放射光



写真H　マッピング爆発球



張りつける画像はMATIERのにじみペンで作成

「炎」の表現に挑戦

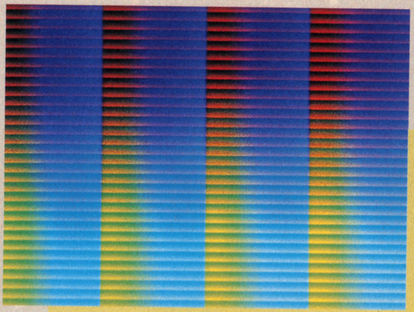


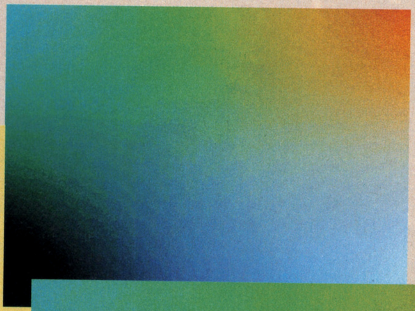
焚火のゆらめきをアニメーションさせせよう



写真L
エアブラシで描き，EPA2で光らせる

X-BASICとグラフィック



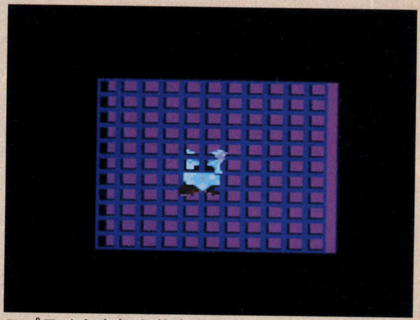




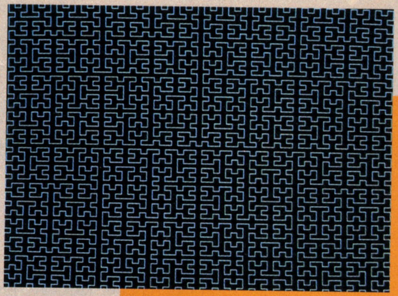
まずは，グラフィックの基礎として X68000で扱える色の話から始めてみよう。

X-BASICではどのような方法で色が決められるのか，X68000で扱える65536色モードでどれだけの表現ができるか考えていく。

そして最後にはちょっとした応用として，ごく基本的な誤差拡散法プログラムを紹介しながら，多色表現にも挑戦している



スプライトを扱う基本を学ぶためのサンプル

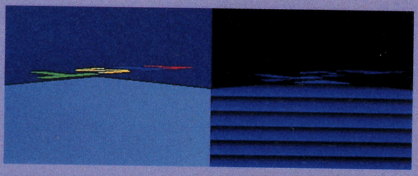


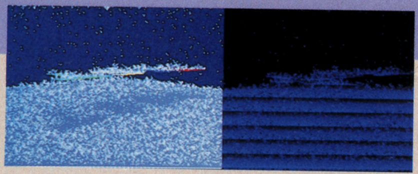


ランレングスの探索パスをヒルベルト曲線に変えることで，より効率のよい圧縮プログラムを目指してみる

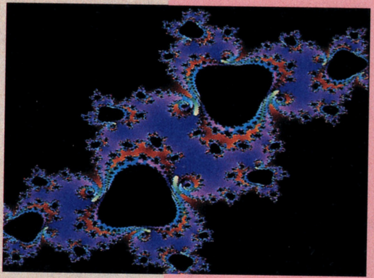


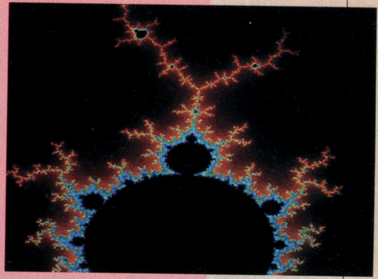


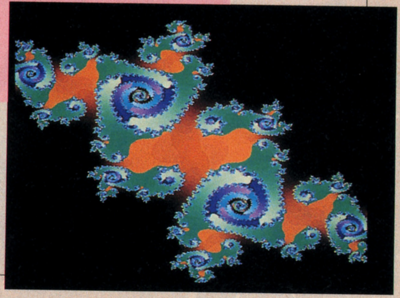




丹氏お得意の画像エフェクタプログラム。今回は，ブラウン運動のシミュレートもどきを使った降雪シミュレータだ。基本的に，2D画像に対するエフェクトしかサポートしていないが，Zバッファを利用することにより3Dへ拡張することもそれほど難しくない

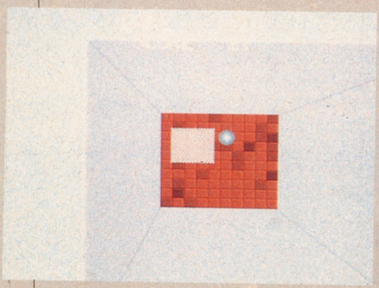


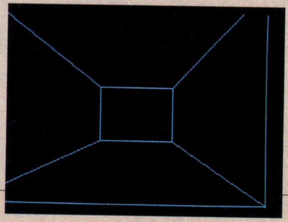




ここでは代表的なフラクタル図形の描画プログラムを紹介している。いろいろな数値を与えるたびに変化するフラクタル図形。単純な数式から導き出されるグラフィックをひとしきり堪能したら，アルゴリズムに目を向けてみよう。

また，最大のポイントである数式をプログラム化するための手法をぜひ学んでほしい





投稿されたショートプログラムの内部を覗いてみる。実際のプログラムでは，X-BASICがどのように使われているかを解説している

[第33回]

コピーのようなもの

江口　響子

響子inCGわ〜るど

先日，藤子不二雄Ⓐ先生のトークショーを拝聴しました。面白い話をたくさんしてくださったのですが，特に印象的だったのは，学生時代に手塚治虫先生と知り合ったいきさつでした。もしかしたら，本などに出ているエピソードなのかもしれません。すでにご存じの方もいらっしゃるかもしれませんね。

きっかけは，手塚治虫作「新宝島」という１冊のマンガでした。「新宝島」は，最初の数ページは吹き出しのない，まるで映画のシーンのような構成になっています。現在では，ごくあたりまえの手法ですが，日本で当時，誰もそのようなマンガを描いた作家はいませんでした。まだ学生の藤子不二雄Ⓐ，藤子Ｆ不二雄の両先生はショックを受け，もうこれはファンレターを出すしかないと思ったのです。

マンガ家を志していた人たちにとって，すでに，手塚先生は神様のような存在でした。きっと，ファンレターは山ほどくるに違いない，確実に読んでもらうためには目立つのが一番だ，と藤子先生たちは考えました。で，手紙に付録をつけることにしたのです。それもただの付録じゃつまらないから，手塚先生の似顔絵を油絵で描いて，額に収めたものにしようじゃないかと話がまとまりました。

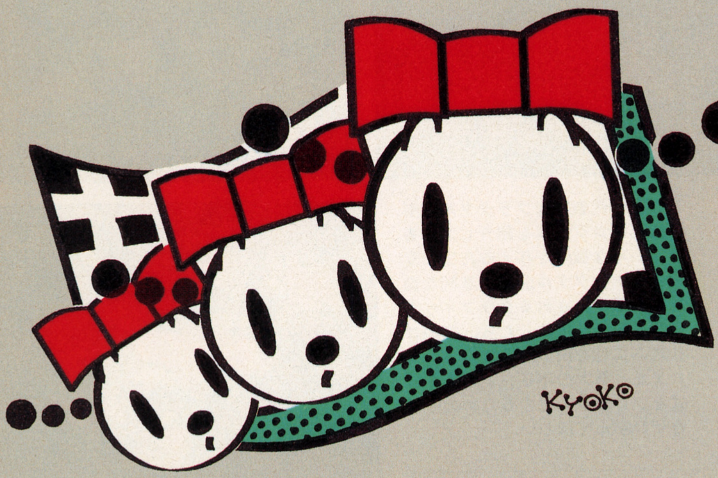
一度も会ったことのない，写真ですら見たことのない手塚先生の似顔絵を，これまたはじめて油絵で描こうというのです。ずいぶん，大胆な思いつきでしたが，それでも，油絵入門の本と油絵の道具を買って勉強し，マンガのかたすみにちょこんと出てくる，「丸い鼻，丸メガネにベレー帽」の似顔絵を手がかりに，あとは想像で手塚先生の顔のようなものを描き上げて送りました。

返事はすぐにきました。その後，何年か手紙でやりとりをしたのち上京し，しばらくしてからあの有名なトキワ荘へ，それも手塚先生が引っ越しをしたあとに入ったのです。

余談ですが，藤子先生たちの最初のころのペンネームに，足塚治というのがありました。マンガの神様，手塚治虫先生にあやかろうとしたそうです。頭塚治という案も出ましたが，頭は手より上についているので，おこがましいのではということになり，それじゃ足にしよう，足なら手より下についているので，失礼にならないだろうと考えたのでした。

藤子マンガのヒット作は，「ドラえもん」，「オバケのＱ太郎」，「パーマン」，「忍者ハットリくん」……と挙げればきりがありませんが，どのマンガにも，本物そっくりに化けることのできるキャラクターが登場します。そのものずばりは，「パーマン」に出てくるコピーロボットです。主人公がパーマンになっている間，代わりを務めるのがコピーロボットで，その黒い鼻を押すと，押した人そっくりの外見に変身します。ただ，中身は，本人よりもずっと優秀な善人になってしまうところがミソで，そこからさまざまな食い違いが生じ，話が展開していきます。

哲学上，厳密な意味でのコピーとは，本物とまったく同じものをいいます。色，形はもちろんのこと，性質，組成などの目にみえない中身も同じでなければなりません。





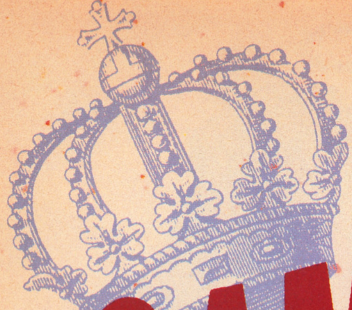
一方，同じように見えるけれど，本物とはどこかしら違っているものを，シミュラクラと呼んでいます。ですから，クローン人間はコピーですが，「パーマン」のコピーロボットは，本当はシミュラクラロボットということになるのでしょう。

さて，コピーはCGならではの表現方法です。手で，同じ絵は2度と描けません。同一キャラクターが多数登場するシーンでは，便利このうえありません。しかし，同じようでありながら，よく見ると1点ずつ違う，手描きの味わいも捨てがたいのです。手描きをとるか，それとも，コンピュータで絵を描く最大のメリットのひとつである，コピーを活用して制作を効率化するかは，作り手としていつも判断に迷うところです。

今回のCGデータ

1280×1024ピクセル
1670万色フルカラーを4×5ポジで出力
総物体数　193　うちメタボール数　48　　光源　2
使用ソフトは，C-TRACE
テクスチャマッピング作成には，MATIER
バンプマッピング作成には，Z's STAFF PRO-68K

影になる部分は，バンプマッピングがつぶれてしまうので，補助光として弱い光を別の角度から当てています。



1993年度

GAME OF THE YEAR

ノミネート作品発表

1月 / 2月 / 3月 / 4月 / 5月 / 6月

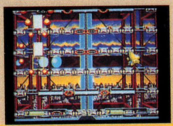
ACTION

●チェルノブ



横スクロール型アクション。ちょっとアブナイ世界が舞台。ビデオゲームアンソロジー2。電波新聞社。3月号。

●ストライクレンジ

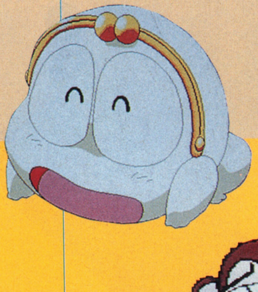


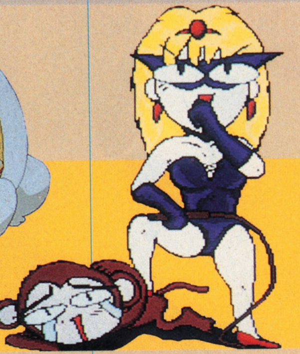
対戦格闘ゲーム。高層スタジアムを舞台にロボットが闘う。ブラザー工業(TAKERU)。4月号。

●エトワールプリンセス



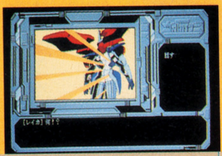
キャラクターが可愛いアクションRPG。リルルは世界を救って玉の輿を狙うのだ。エグザクト。5月号。





ADVENTURE

●機甲装神ヴァルカイザー



◀アニメファン好みのADV。美少女とメカがアニメーションする。ブラザー工業(TAKERU)。開発はサイレンス。2月号。

▶アドベンチャー的雰囲気を持った近未来が舞台のアクションRPG。戦闘はコマンド選択式。ブラザー工業(TAKERU)。7月号。

ROLE PLAYING GAME

●ドラゴンスレイヤー英雄伝説



6章で構成された「ドラゴンクエスト」タイプのRPG。剣と魔法の世界を旅するセリオス王子。SPS。2月号。

●幻影都市



●ヴェルスナーグ戦乱



オートバトルや各種の仕掛けが盛り込まれた新機軸RPG。主人公ザレクはモンスターを倒す賞金稼ぎ。ファミリーソフト。7月号。

SHOOTING

●究極タイガー



ヘリコプターを操作して敵を撃破する。縦スクロール型シューティング。KANEKO。3月号。

▶とにかく熱くなるまで連射。アーケードからの移植作。電波新聞社。ビデオゲームアンソロジーシリーズ3。4月号。

●スターフォース



●KU²

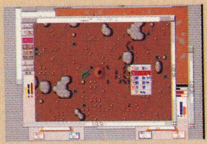


◀敵を食ってエネルギーにして攻撃する。ユニークなシューティング。パンサーソフトウェア。5月号。



SIMULATION

●シムアント



気分はすっかりアリ，の生態系SLG。SX-WINDOW上で動作する。イマジニア。3，4月号。

●メガロマニア



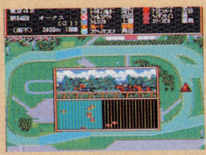
種族を操り新しい惑星を我がモノに！　イマジニア。5月号。

●信長の野望・覇王伝



シリーズ第5作，戦国大名になって全国を統一するSLG。光栄。6月号。

●Winning Post



馬主の気分が味わえる競馬SLG。資金1億円を元手に凱旋門賞を狙え！　光栄。8月号。

●銀河英雄伝説III



同名小説が原作の，宇宙を舞台にした陣取り型戦略SLG。ブラザー工業(TAKERU)。7月号。

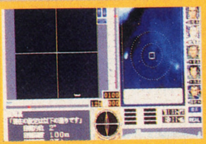
●蒼き狼と白き牝鹿・元朝秘史

舞台はモンゴルとユーラシア大陸。チンギス・ハーンは戦いと子作りで帝国建設を目指す。光栄。4月号。



▶潜水艦を操るSLG。同名の漫画が原作。シナリオは11種類。ジー・エー・エム。6月号。

●沈黙の艦隊



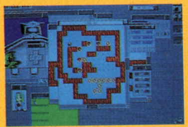
●大航海時代II



▶中世の海洋を舞台に6人の主人公が冒険を繰り広げる。光栄。7月号。

PUZZLE/TABLE GAME

●倉庫番リベンジSX-68K



倉庫の荷物を指定位置へ運ぶパズルゲームをSX-WINDOW上で再現。シャープ。開発はシンキングラビット。5月号。

●極



▶思考時間も短くてとにかく強い将棋ソフト。ログ。2月号。定跡集(PC-98版と共有)も発売中。

ET CETERA



●リブルラブル



リブルとラブルで敵や宝箱をパシし奇跡を起こす。電波新聞社。アンソロジー4。7，8月号

●The World of X68000

第1回全日本X68000芸術祭での優秀ゲーム作品5本を収録。電波新聞社。10月号。



さあ，今年もGAME OF THE YEARを選ぶときがやってまいりました。1993年，あなたのゲームライフはいかがでしたか。

X68000版の新作ゲームは，昨年や一昨年に比べると発売タイトル数は残念ながらやや減少してしまいました。そのうちOh!X誌上で紹介したものは41作です。タイトル数の減少は，他機種に比べてユーザー数が少なく，1タイトルあたりの販売本数が見込めないとか，X68000には「目の肥えた」ユーザーが多くてよほどきちんとしたものでないと売れない，などというメーカーの見識が反映されているようです。

しかし，そのような状況ではありますが，振り返ってみると，結果的にはX68000ゲーム市場は決して元気をなくしているわけではなく，むしろ盛況だったといってよいと思われます。タイトル数こそ少なめながらも，ゲームデザイン，グラフィックや音楽などによる演出，そのほかいろいろな点で作品の質が高いものが発表されています。また，他機種パソコンとは毛色が違うといわれるユーザーの好みをきちんと捉えた，X68000らしい作品が増えるなど，たいへん喜ばしいことだといえるでしょう。

さて，そんななかで選ばれるGAME OF THE YEAR。その栄冠に輝くのは果たしてどのゲームでしょうか。1年間を振り返って，あなたをいちばん楽しませてくれた作品。それがあなたの1票を待っています。

ACTION / ADVENTURE / ROLE PLAYING GAME / SHOOTING / SIMULATION / PUZZLE/TABLE GAME / ET CETERA

7月

●餓狼伝説



対戦格闘ゲーム。テリーとアンディ兄弟が世界一の格闘家を目指す。魔法株式会社。6，8月号。

●悪魔城ドラキュラ

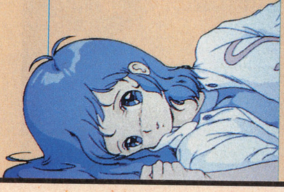


コナミの名作のX68000オリジナル版。横スクロール型。凝った演出が人気。コナミ。7～9月号。

●ロボットコンストラクションR.C.



自分で設計したロボットの動きをプログラミングして闘わせる。エレクトリックシープ。8月号。



8月

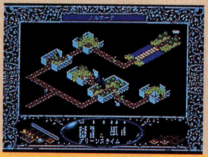
●クレイジークライマー/クレイジークライマー2



●宝魔ハンターライム



●ダーク・オデッセイ



フィールド型RPG。契約した仲間を召還しながら戦いを繰り広げる。ソフトプラン。9月号。

●レッスルエンジェルス

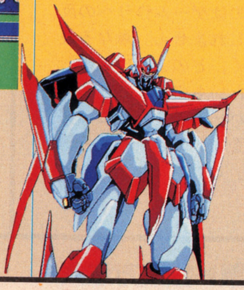


カードを引いて技を繰り出す格闘ゲーム。目指すは女子プロレスの世界一。ブラザー工業(TAKERU)。開発はグレイト。9月号。

●あにまーじゃんV3



性格も容姿も三様の3人姉妹と対戦する麻雀ゲーム。サイレンス。開発はソニア。10月号。



9月

◀2本のジョイスティックでビルの壁を登る。電波新聞社。アンソロジーシリーズ5。9月号。

◀美少女アニメノリのADV。1話ずつ発売。ブラザー工業(TAKERU)。開発はサイレンス。8月号。

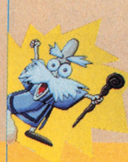


●コットン



可愛い魔法使いコットンは「WILLOW」欲しさに魔物退治。横スクロール型。EAビクター。9，10月号。

10月

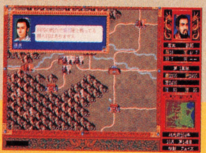




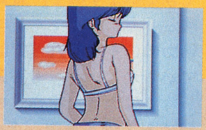
●ダイアット・ヴァークス



●項劉記



●スーパーリアル麻雀PII&PIII



アーケード版ではめちゃ強だった麻雀ゲームの移植作。3人娘と対戦。ビング。12月号。

11月

●ストリートファイターIIダッシュ



いわずと知れた大人気格闘ゲームの本家。カプコン1年ぶりの待望の移植作。カプコン。1994年1月号。

●ぶたさん



可愛いぶたさんの爆弾投げ。電波新聞社。アンソロジー6。11月号。

●ネメシス'90改



MSX版「グラディウス2」の移植作。グラディウス帝国を救え！　横スクロール型。SPS。12月号。

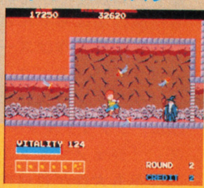
◀人類と獣人が共存するファンタジー世界で異変が起こる。ブラザー工業(TAKERU)。11月号。

◀項羽になるか劉邦か。時は紀元前200年。古代中国の覇権をめぐる闘いのSLG。光栄。12月号。

▶石板の色，模様を組み合わせて消すパズル。電波新聞社。開発はサクセス。1994年2月号。

12月

●ドラゴンバスター

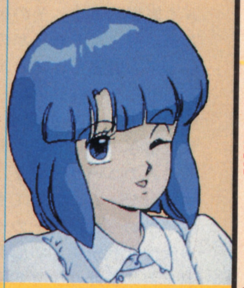


剣を使ってドラゴン退治。お姫様や魔法使いも登場。電波新聞社。アンソロジー7。1994年1月号。

●餓狼伝説2



対戦格闘ゲーム型。2ラインバトルが特徴でX68000版は四天王もプレイ可。魔法株式会社。1994年1月号。



●キーパー



●X68000ログインソフトウェアコンテスト傑作ゲーム選



ログイン誌上で行われているコンテストの入選作12本を収録。アマチュアならではの多彩なゲーム。アスキー。1994年1月号。

さあ，あなたも１票を！

毎年，読者の投票によって選ばれるGAME OF THE YEAR。例年どおり，選択応募部門と自由応募部門に分けて投票していただきます。

選択応募部門

選択応募部門は，５つの部門に分けてそれぞれの該当作品を１つ決定します。投票は原則として今月号の読者アンケートはがきに記入していただきます。１つの作品をいくつかの賞に推薦してもかまいません。

各部門賞は，１投票につきそれぞれ１点として集計しますので「思い入れ度」は残念ながらここには反映されません。ここではあくまで多くの人に支持されたかどうかが決め手となります。もちろん，Oh!X2月号を10冊購入すれば，はがきを10枚投票することが可能ですが，集計担当者はたぶん「あ，おんなじ人だ」と気づくでしょう。その場合どうするかはまだ決めていません。ちなみに，ほかの号のはがきに書いても無効とします。

投票の対象となる作品については，今年はいままでと方針を変更します。これまでは，各賞についてそれぞれノミネート作品を編集室で選んでいましたが，今年はどの賞についても対象作品は，

1993年に発売された新作ゲームのうち，
Oh!Xで紹介した全作品

とします。具体的な作品名と発売時期については前ページで紹介したとおりです。レビューを掲載した号も書いてありますので，ゲーム内容など詳しいことはバックナンバーを見てくださいね。

Oh!Xゲーム大賞

なんといっても「大賞」です。これこそが今年のX68000ゲーム界を代表するものだ！ という作品を選んでください。

とはいえ，ゲームへの思い入れや楽しみ方は人それぞれ。
「遊んだ時間がいちばん長かった」
「初めて遊んだジャンルだけどハマった」
「誰がなんていおうと，コレがいちばん」
どんな観点で選ぶも自由です。ただし選べるのはひとつだけなので，好きなゲームがたくさんある人はじっくり悩みましょう。即決の人も悩んだ人も推薦理由はちゃんと書いてくださいね。

昨年の受賞作はズームのＦ１レースゲーム「オーバーテイク」でした。２位は僅差で「グラディウスII」。以下，得票の多い順に「ファイナルファイト」「スターウォーズ」「ジェノサイド２」と続きました。

今年も大作が目白押しですが，栄えある大賞に輝くのはどの作品でしょうか？

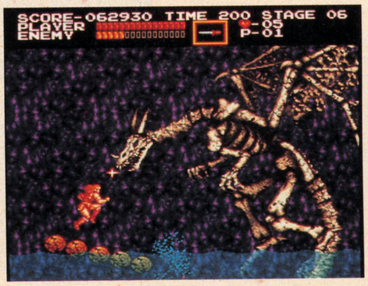


グラフィック賞

ゲームの第一印象を決定するのはやはり，何といっても見栄えがするかどうかなのかもしれません。グラフィックはそのゲームの世界を作り上げる最も重要な要素のひとつです。特に，X68000らしさという観点でゲームを語るのならば，グラフィックを抜きにすることはできません。今年も多くのゲームが，X68000のもつグラフィックパワーを活かした素晴らしい演出で私たちの目を楽しませてくれました。

さて，そのグラフィックの表現力や完成度を評価するのがこの「グラフィック賞」部門です。全体的な完成度はもちろんですが，細かいところでの凝った演出や緻密なデザイン構成，独特の色使いなど，チェックポイントはたくさんあると思います。どういう点を高く評価するかはそれぞれ好みが分かれるところでしょうか。

昨年選ばれたのはズームの「ジェノサイド２」。２位以下を大きく引き離しての受賞でした。



音楽賞

ゲーム世界の演出において，グラフィックと並んで重要なのが音楽です。音楽や効果音の善し悪しはゲーム全体の雰囲気に大きく影響します。そのゲームにのめり込めるか，醒めてしまうかの鍵を握っているといっても過言ではありません。

しかし，X68000ユーザーの音楽環境は非常にまちまちです。MIDIの普及率の増加は目ざましいものがありますが，内蔵音源のみのユーザーもまだ過半数を占めています。ゲームにおいても各種の環境への対応が求められ，メーカーにとってはなかなか大変な状況です。しかし，そのなかでそれぞれのクオリティの高さを実現することは評価の高さにつながるでしょう。

昨年はコナミの「出たな!!ツインビー」が受賞。「オーバーテイク」「グラディウスII」も健闘し，得票差の少ないところで戦いが繰り広げられました。





プログラミング技術賞

技術力というのは，ゲームを遊ぶうえでは必ずしも前面に出てくるものではありませんが，これの充実がゲームの根本を支えているといってよいでしょう。現在，技術力そのものは全体的に上がってきていますので，重要なのはゲーム作りへの反映させ方になってくるかもしれません。単なる技術力のあるなしだけではなく，どのように駆使しているかという点も考慮したいものです。

昨年は大賞受賞作「オーバーテイク」がこの部門でも１位を獲得。２，３位は「ストライダー飛竜」「ファイナルファイト」とカプコンが力をみせつけました。

ゲームデザイン賞

優れたコンセプトで高いゲーム性をもつ作品，という観点での評価を行うのがこの部門です。企画やデザインとは，単なるひとつの思いつきの面白さだけではありません。たくさんのアイデアをうまくまとめてバランスをとり，ひとつの作品として完成させてこそ，そのゲームデザインの新しさやユニークさが光ってくるのでしょう。

ゲームの部分部分の面白さでみせるか，ゲーム全体を貫く一貫した思想や世界観でユーザーの心をつかむか，方法はさまざまですが，ゲームデザインの基本設計がきちんとなされてこそ最終的な完成度の高さが実現できるに違いありません。

昨年は１位がイマジニアのパズルゲーム「レミングス」，２位は「ポピュラスII」と，海外ソフト移植作品が上位を占めました。



選択応募部門への投票▶投票方法A

自由応募部門

選択応募部門では作品への思い入れを表現できずに不満がくすぶっているそこのアナタ！
その熱い思いはこちらの自由応募部門にぶつけてください。応募形式は自由ですが，自由応募部門の○○（例：勝手にGAME OF THE YEAR△△賞）と明記してください。パワーあふれる応募をお待ちしています。

主演・助演キャラクター賞

毎年恒例のキャラクター賞です。

昨年は，誰が主役だ？　というゲームが多かったのか主演・助演が混沌とした投票結果になったため，え〜い面倒だ，まとめてやっちゃえ，ということで選ばれたキャラクターで主演と助演の区別はなしとしました。一番人気は「レミングス」の爆死するネズミたち。それを追う「オーバーテイク」に登場のご存じズームネコ。あとはオッサンやら肉やらガムやら，もう何がなにやら……。

さて，1993年のゲームのキャラクターはカッコいいヒーローあり，ユニークな動物ありとなかなか多彩な個性の持ち主揃いです。可愛い女の子もたくさんいるので，投票率の急上昇も期待される部門であります。

勝手に GAME OF THE YEAR

選択応募部門の賞にはちょっとあてはまらないけれど，何か賞をあげたいあのゲーム。決して大作とはいえないけれど，なぜかこころ惹かれるあのゲーム。誰にもそんなゲームがありますよね。

この「勝手にGAME OF THE YEAR」は，勝手にどんどん賞を作って，楽しませてくれた作品にあげちゃおう，というコーナーです。おちゃらけでも，しくしくでも，ぷんすかでも，にやりでも，なんでもありの無法地帯！



底抜け脱線ゲーム体験談

うまい人はよりハイレベルを目指し，下手な人もそれなりにと，同じゲームでも楽しみ方は人によりけりです。また，ひとりでやるのと誰かと遊ぶのでは燃え方も全然違いますよね。プレイしているうちに，思いもよらない楽しみ方を発見したりすることも……。たとえ本来の目的から外れたって，いいじゃないの幸せならば。

そこで，あなたなりのゲームの遊び方や「こんなことが起きちゃった」というハプニング報告など，楽しい体験談を募集します。笑いを呼ぶもの，涙を誘うもの，なんでもかまいません。あなたのユニークで貴重な体験を，ほかの読者とともに分かち合おうではありませんか。

イラスト

大好きなゲームだけど，口下手でその気持ちがうまくいえない……そんなアナタはそのゲームに対する情熱や愛情をイラストで表現してみませんか。最愛のキャラクターを描くもよし。プレイ報告を図解説明するもよし。次回作への期待と希望を込めるもよし。プレイの合間の息抜きに，楽しませてくれるそのゲームへのラブレターでも書くように，イラストを描いて送ってくださいね。

イラストの投稿▶投票方法C



読者レビュー

Oh!Xのゲームレビューでは，新作ゲームを実際にプレイして紹介しています。が，より早い情報をお届けするためには，新作発売前後の短い日程のなかでのテストプレイ，限られた誌面，雑誌制作の締め切りとの戦いなど諸般のかなしい事情ゆえ，そのゲームのすべてを説明することはなかなかできません。

そこで，レビュー内容に異議アリの人，もっと突っ込みたい人，華麗なプレイを自慢したい人などのゲームレポートを募集します。

そのほか

常日頃，ゲームについては一家言ある，という人はいませんか。胸のうちではゲームへの熱情が渦巻いているのに，それをぶつけるものがなく爆発寸前の人はいませんか。

そんな人もやさしく(？)受けとめてあげましょう。何か形にしたものを送ってみてください。

ゲームに関することならば，どんな形式でもかまいません。個々のゲームについてだけではなく，1993年のゲーム全体についての感想でも，メーカーへの要望でも，好きなジャンルのゲームの話でも，疑問に思っていることでも，テーマは自由です。グローバルな視点でのレポート，個人的な主張など，なんでも大歓迎です。

自由応募部門への投票▶投票方法B

1993 GAME OF THE YEAR 投票方法

投票方法A　選択応募部門への投票

■アンケートはがき　記入の仕方は去年と同じです。

1）　Oh!Xゲーム大賞に推薦する作品名を書く
2）　右の欄にOh!Xゲーム大賞の推薦理由を書く
3）　1.〜4.の欄にそれぞれの賞に推薦する作品名を書く
4）　右の欄の（　）のなかに，推薦したい賞の番号（1〜4）を書く
5）　その推薦理由を書く

Oh!Xゲーム大賞　1)	ゲーム大賞推薦理由　2)
1.	
2.　3)	推薦理由（　　）　4)　5)
3.	
4.	

これで完了。

でも，なんかもの足りないぞ，という人▶投票方法B

投票方法B　・自由応募部門への投票　・もっといいたいぞ，という人

■アンケートはがきの編集室へのメッセージ欄
■官製はがきまたは封書
書式は自由。フロッピーディスクでも可ですが，お送りいただいたものはご返却できません。

投票方法C　イラスト

■官製はがきまたは封書
いつものイラスト投稿と同じです。汚れないように注意してください。できればはがきサイズで，モノクロのみ。あんまり巨大なものはたぶんボツになるでしょう。

宛先

〒103 東京都中央区日本橋浜町3-42-3
ソフトバンク株式会社
Oh!X編集部
「1993年度GAME OF THE YEAR」○○○係

締め切り　1994年2月18日必着

新製品紹介

ハイパーピクセルワークスver.2.00

Takahashi Tetsushi 高橋 哲史

「ハイパーピクセルワークスver.2.00」(以下HPW)は，ハイパーが開発したグラフィックツールです。その前身にあたる「ピクセルワークス」は同人ソフトとして世に出て，一部ユーザーからの好評を博していました（私は今回初めて使用するのですが……)。強力なエフェクト機能をウリにしたこのHPW。マニュアルに堂々と「初心者向けではない」と明記するという，その実力のほどはいかがなものでしょうか。

◆ 描画機能 ◆

さて，エフェクトがウリといってもHPWは単なる画像加工ソフトではありません。描画機能もきちんとあって，一から絵を描くことができるようになっています。

トップウィンドウにはフリーライン，コネクトライン，ベジェ曲線，ボックス，サークルなど一般的なものが過不足なく揃っています。もちろんペイントや閉曲線ペイント，スプレーなどもしっかりついています。

ここでHPWの大きな特徴は「すべての描画機能がアンチエリアシングをサポートしている」ということです。つまり「初めからギザギザ（ジャギー）のない滑らかな線を引くこと」ができるのです。またペイントなどもそうしたアンチエリアシングな線で囲まれた領域を塗りつぶせるようになっています(写真1)。これはとても素晴らしいことなのですが，フリーラインなどで引いた線が，私には「アンチエリアスされた線」というよりは「ふちのぼけたペンで引かれた線」のように見えてしまうのです。これは私個人の感覚なので意見の分かれるところだとは思いますが……。個人的には，アンチエリアスの深さを設定できるようにすればよかったのではないかと思います（現在は固定)。

とはいっても，アンチエリアス対応のペイントルーチンはかなり重宝します。なんといっても多階調で取り込んだ絵をそのままペイントできるのですから（写真2)。ま，これももちろん「ゴミが出ないように綺麗に取り込めるスキャナ」をお持ちの方に限られる特典になってしまいますが（私はいつも取り込んだあとせっせとゴミとりに励んでいます。それがCG道だだだっ)。

ちなみにHPWがサポートしているスキャナは，GT-6000とHS7Rシリーズのみです。しかしスキャナ読み込みは外部プログラム呼び出しの形をとっているので，順次対応機種を増やしていくことは簡単だと思います。ここで特筆しておきたいのは，本来なら2値取り込みしかできないHS7Rシリーズで，グレイスケールの取り込みを可能にしていることです(私はHS7RIIユーザーなのです)。個人的には，グレイスケールで取り込めるようになったことよりも，Z'sなどより広い範囲の取り込みが可能になったことのほうが嬉しいのですが。

また，すべての描画機能において裏画面を描画色として使用できるようになっています(裏画面こすり出し)。またマスク機能も充実していて，マスクペイントはもちろん指定色（赤色系統といった感じの広範囲指定も可）のマスク変換などもサポートしています。マスク情報だけのロード/セーブも可能です。またグラデーションも内部24bit演算により綺麗にディザリングされています。

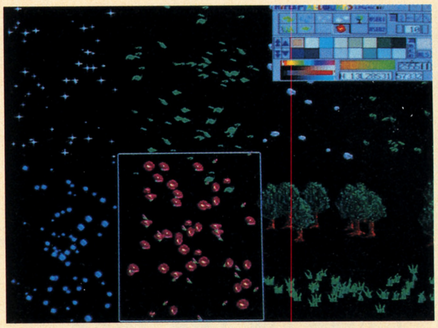


写真3　プリセットのオブジェクトを散らしてみた

ほかに変わった描画機能としては「ランダムオブジェクト」の存在が挙げられます（写真3)。これは，木の葉などパターン化されたもの（オブジェクト）を指定範囲にランダムに散らす機能です。もちろん透明色の設定もできるので，各オブジェクトは奇麗に重なり合います。使いようによっては作業の効率アップなどが望める機能でしょう。

最後に少々気になる点をひとつ。それはダブルクリックの扱いです。通常ダブルクリックは「クリックの間隔」によって検出されますが，HPWでは「間隔にかかわらず同じドット上でクリックされたとき」をダブルクリックとして扱います。なぜこのようにほかのソフトとは異なる仕様を採用したのかいまひとつ理解に苦しむところですが，これにより各描画機能を使っている最中にとまどうことが多々あります。

◆ エフェクト・合成機能 ◆

さて，HPW最大のウリであるのがこのエフェクト機能です。波状変形やモーションブラー，放射光など既存のグラフィックツールにはない，使えるエフェクトが満載されています（写真4,5)。Z's_EXにあったフレアや微分処理，パース変形などももちろんサポートされています（いま気がつきましたけど，モザイクやフラクタル処理はないようですね)。それぞれ効果の深さや向きなどが細かく設定できるようになっているので，思い通りの効果が得られると思い

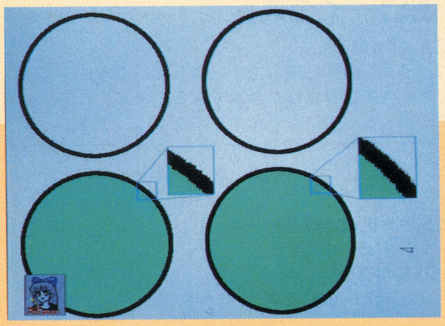


写真1　左がノーマル，右がアンチエリアスONの状態

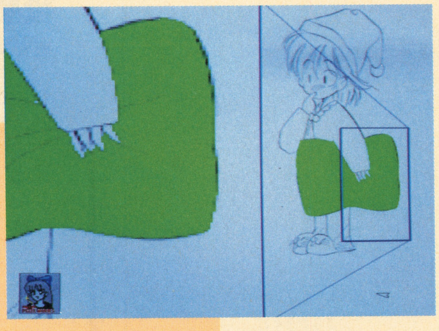


写真2　CZ-8NSIで取り込み後，直接ペイント（取り込みはMATIER)



写真 6　はめこむ画像を裏画面に転送



写真 7　左上のフキダシ(?)にはめこむ



写真 8　マルチマッピング完了

ます。ただここで私が不満なのは，パラメータの操作にオートリピートが効かないということです。つまり１から32にパラメータを変更したいときは31回もマウスをクリックしなければならないのです。これは勘弁してほしいところです。

またHPWは裏画面をひとつ持っており，表画面との合成が行えるようになっています。通常の合成に加え，グラデつき合成なども簡単にできるのが特徴です。あと私が面白いと感じたのは，表画面の任意形に裏画面をはめこむマルチマッピングです（写真6,7,8)。「領域の形状から疑似的な3D計算を行い，効果的なマッピングを行う」とのことですが，効果は写真でご覧の通りです。使い方次第ではいろいろと面白いことができるのではないかと思います。

◆ エトセトラ…… ◆

HPWはメインメモリ2Mバイトでも動作します。というか，2Mバイトあればそれで十分なのです。MATIERのようにメモリがあればあるほど裏画面がいっぱい使えるようになるなどということはありません。2Mバイトでメモリが足りなくなる場合というのは，よほど大量のメモリを消費する外部プログラムを呼び出したときだけです（デフォルトではそんな「鬼」な外部プログラムは登録されていませんが)。

また正式な動作を保証されたわけではないのですが，384×256，256×256ドットモードも備わっています。ゲームのグラフィックを描く人には便利な機能ではないでしょうか？　加えて，見るだけなら正方形ドットモードもサポートされています（描画はできずに本当に「見るだけ」)。さらに，ゲームに使用するというのに関連したことでは，動画機能の存在も挙げられます。これは裏画面に動画パターンを置いて表画面の任意の位置に表示する機能です（DōGAとはまったく使用用途が違います)。主に目パチ口パクの確認に使うといったところでしょうか?

あとHPWのルーペはなかなか賢く使えて嬉しいです(写真９)。全画面ルーペなのですが，1/1ウィンドウ（実サイズの画面）がエディットしながら確認できるようになっています。

それとこれは余談ですが，どうもPICのロードが遅いようです。同じファイルをMATIERやコマンドライン上から読み込んでみましたが，3割増しくらいの感覚でHPWのほうが時間がかかります。読み込み時にそんな特殊なことをしているとも思えないのですが……。小さいPICならあまり気にならないのですが，100Kバイトを超えるような巨大PICだと少しストレスがたまってしまうような気がします。

◆ 最後に ◆

HPWは，いままでのグラフィックツールにないさまざまな試みのなされているツールだといえます。が，同時にかなりクセの強いツールであることも間違いありません。前述のダブルクリックの扱いもそうですが，ほかにも「画面からメニューを消すことは絶対にできない（最小化したアイコンが画面上に残る)」とか「ファイルウィンドウの仕様が特殊で，慣れるまでとまどう」など気になる点が多々あります。このあたりはどうも制作者側の個人的な主張を押しつけられているようで，あまり気持ちのよいものではありません。世の中にはすでに，ある程度スタンダードな操作体系がお約束としてあるのですから，相当なメリットがない限り，それ以外のものを採用するのは百害あって一利なしだと思います。使いでのある機能が揃っているだけに，これは残念なことです。

といってもいま現在すでにZ's STAFFやMATIERを所有している人でも，新たに買って損をするということはないといえるソフトです。HPWでしか実現できないこと（あるいはHPWのほうが手軽にできること）が結構あるのも事実なのです。ただ，初めてのツールとしては前述のこともあってどうかなとも思いますが，もしHPWだけを使うというのであれば，それは別に差し支えないと思います。値段も，数あるグラフィックツールのなかではかなりお安いですしね。

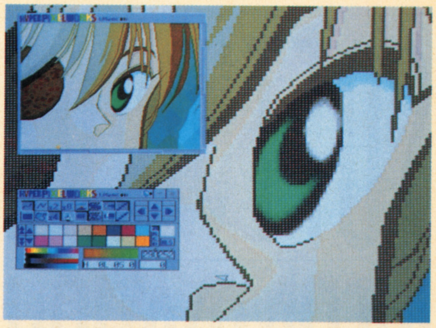


写真 9　左上に出ているのが1/1ウィンドウ

ハイパーピクセルワークス
X68000用　5"2HD版　19,800円(税込)
ブラザー工業（TAKERU)　☎052(824)2493

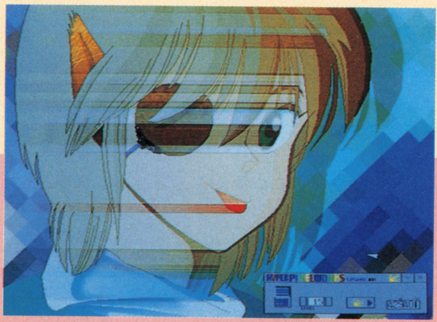


写真 4　放射光を顔の右はじから画面の左はじに設定

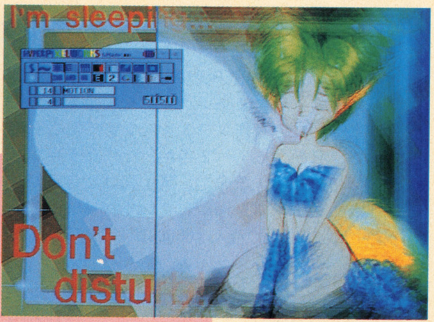
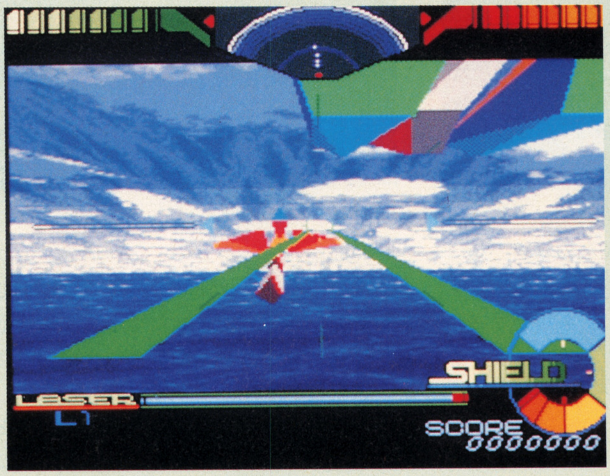
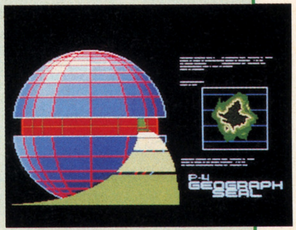


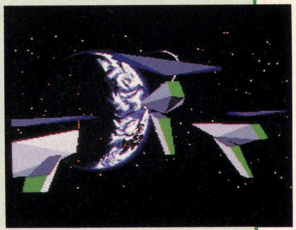
写真 5　モーションブラーを使って動きを出してみる

SOFTWARE INFORMATION

年も変わって受験生の人には試練の季節となりました。苦しいことも，すでに乗り越えた人にとっては懐かしい思い出かもしれませんね。がんばるなかでも息抜きは必要。ゲームも新作がアナタを待っています。







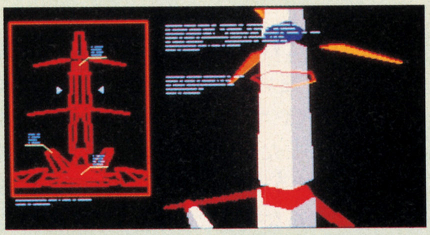
ジオグラフシール

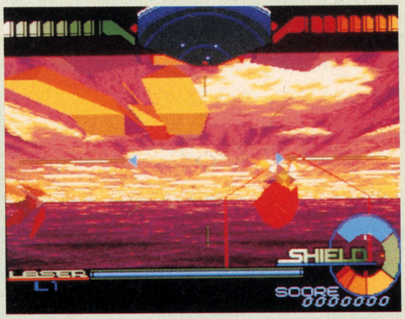
あの「エトワールプリンセス」のエグザクトの新作「ジオグラフシール」。キャラクターの個性を前面に出した前作とは雰囲気をがらりと変え，今度の作品はポリゴンを駆使したオリジナルアクションシューティングゲームである。

敵の惑星WS090の極点に建造中の地上プラント兵器を破壊せよ，という命令が「プロジェクト・ジオグラフシール」。プレイヤーは制空圏外から惑星に侵入し，極点を目指す。都市戦あり，海上戦ありで，強制スクロール，空気遠近法採用，ダンジョンなど，ステージごとに構成が異なるため，「ボスを倒す」「ターゲットを破壊する」などクリア条件もさまざま。

RS-232Cクロスケーブルでつないでの対戦モードもある。発売は3月上旬の予定。

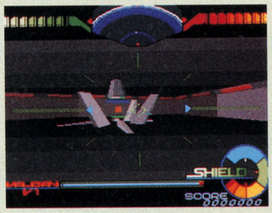
X68000用　5″2HD版　9,800円(税別)
エグザクト　☎025(284)7304

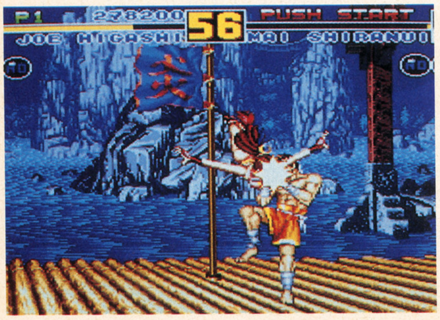












名作文庫ソフト

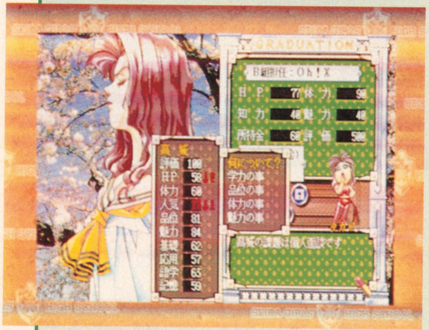
ブラザー工業(TAKERU)では，1994年2月1日より，企画「名作文庫ソフト」を開始する。

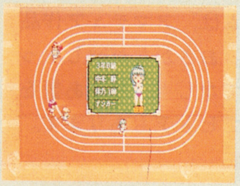
これは過去の名作ソフトを低価格で再販するもので，発売から1～2年たった作品を半額以下の価格で提供する予定。第1弾は2月だが，今後ラインナップを増やしていく。

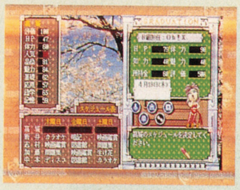
第1回の製品のうちX68000版は以下のとおり。

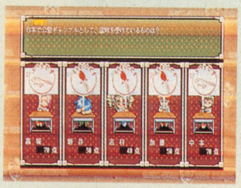
スーパー大戦略	2,000円
大戦略Ⅲ'90	2,500円
ブルトン・レイ	2,500円
ブリッツクリーク	2,500円
麻雀　武蔵	5,800円
Xak	3,900円
ルーンワース「黒衣の貴公子」	2,000円
ファーストクイーンⅡ	2,500円
エメラルドドラゴン	3,900円
バーンウェルト	3,500円
サバッシュ	4,800円
エイトレイクスゴルフクラブ	2,000円
ウォーニング68	3,800円

卒業〜GRADUATION









先月号でグラフィック画面をご紹介した「卒業」。開発は着々と進行中で，編集部にも途中バージョンが届けられた。これで見る限り，先行発売されているPC-9801版とほぼ同じ内容のようである。

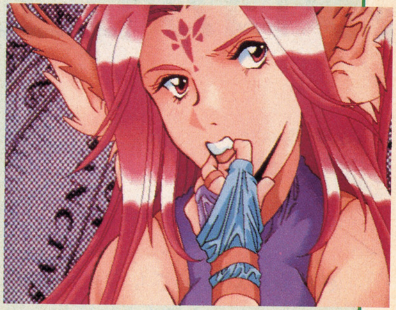
可愛い教え子は5人の高校3年生。卒業までの1年間のさまざまなイベントを乗り越えて彼女たちはどんな春を迎えるか。それぞれの進路に分かれる前の最後の学園生活で，君の「指導力」が試されるのだ。

X68000用　5″2HD版　7,800円(税込)
ブラザー工業(TAKERU)　☎052(824)2493





B-Field!

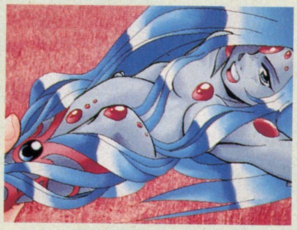


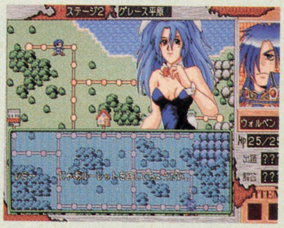
コミカルタッチのクイズゲームの登場である。双六方式でMAPを移動して戦いを繰り広げるのだが，その戦闘がクイズとなっている。問題は一般からゲーム，アニメにいたるまで全部で約500問。勝てば経験値が増え，それにつれてレベルが上がっていく。

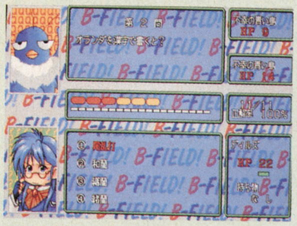
MAPは全部で5面あり，面ごとに女性エルフなどのビジュアルシーンを見ることができる。

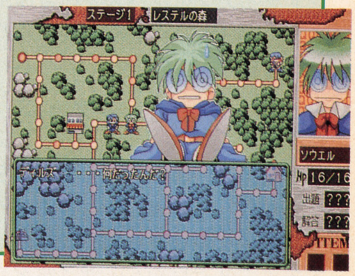
開発は「ダイアットヴァークス」のstudioインフェルノ。前作では人類と獣人が共存するファンタジー世界が構築されていたが，今回はどんな世界を楽しませてくれるのだろうか。発売は1月28日。

X68000用　5″2HD版　3,900円(税込)
ブラザー工業(TAKERU)　☎052(824)2493









写真はPC-9801版です

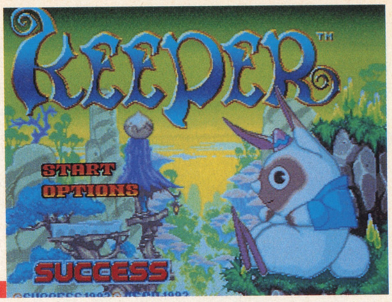
発売中のソフト

★Y300-A ver1.1　マグマソフト
X68000用　5″2HD版　34,800円(税別)
★キーパー　電波新聞社・サクセス　12/22
X68000用　5″2HD版　8,800円(税別)
★餓狼伝説2　魔法株式会社　12/23
X68000用　5″2HD版　9,800円(税別)
★ドラゴンバスター　電波新聞社
X68000用　5″2HD版　5,300円(税別)
★宝魔ハンターライム6　ブラザー工業(TAKERU)　1/10
X68000用　3.5/5″2HD版　1,500円(税込)
★マッドストーカー　ファミリーソフト　1/14
X68000用　5″2HD版　7,800円(税別)

新作情報

★卒業〜GRADUATION　ブラザー工業(TAKERU)　1/20
X68000用　5″2HD版　7,800円(税込)
★B-Field!　ブラザー工業(TAKERU)　1/28
X68000用　5″2HD版　3,900円(税込)
★SX-WINDOW 開発キットWorkroom SX-68K　シャープ
X68000用　3.5/5″2HD版　価格未定
★EGWord SX-68K　シャープ
X68000用　3.5/5″2HD版　価格未定
★SX-WINDOW 開発キット用サポートツール集　シャープ
X68000用　3.5/5″2HD版　価格未定
★宝魔ハンターライム7　ブラザー工業(TAKERU)　2/10
X68000用　3.5/5″2HD版　1,500円(税込)
★麻雀航海記　ブラザー工業(TAKERU)　2/未
X68000用　5″2HD版　5,800円
★ジオグラフシール　エグザクト　3/上
X68000用　5″2HD版　9,800円(税別)
★宝魔ハンターライム8　ブラザー工業(TAKERU)　3/10
X68000用　3.5/5″2HD版　1,500円(税込)
★マージャンクエスト　SPS
X68000用　5″2HD版　価格未定
★ロボスポーツ　イマジニア
X68000用　5″2HD版　価格未定
★Traum　象スタジオ
X68000用　5″2HD版　価格未定
★鮫！ 鮫！ 鮫！　KANEKO
X68000用　5″2HD版　価格未定
★達人　KANEKO
X68000用　5″2HD版　価格未定
★エアバスター　KANEKO
X68000用　5″2HD版　価格未定
★サバッシュII　ポプコムソフト/グローディア
X68000用　5″2HD版　価格未定
★麻雀悟空・天竺への道　シャノアール
X68000用　5″2HD版　9,800円(税別)
★スタークルーザーII　アルシスソフトウェア
X68000用　5″2HD版　価格未定
★ぷよぷよ　SPS
X68000用　5″2HD版　価格未定
★エキサイティングアワー/出世大相撲　電波新聞社
X68000用　5″2HD版　5,300円(税別)
★魔法大作戦　E.A. ビクター
X68000用　5″2HD版　価格未定

プロが仕上げた爽快パズル



Kiyose Eisuke
清瀬 栄介

久々に登場のアクションパズルゲームは，アマチュア作品の商品化という生い立ちも変わり種。おさかなが欲しかったのに巻貝がくるなんて，や〜ん人でなし！　じゃあ蟹をもってきて……。と海のものづくしでヒトデもあります。

そろそろX68000でもパズルゲームをしたいなあ，というアナタに新作の紹介を。サクセスの「キーパー」だ。

このゲーム，もとは先月号で紹介した「X68000傑作ゲーム選」の中の「レーシーズ」というゲームだったもの。アマチュアが考えたゲームを題材にプロがリメイクしたというわけ。しかも作ったのが「コットン」のサクセスっていうんだから，原作者も幸せ者だね。

ちなみに発売は電波新聞社。アマチュアが作ってサクセスが製品化，発売は電波新聞社という夢の(?)トライアングルパスだ。

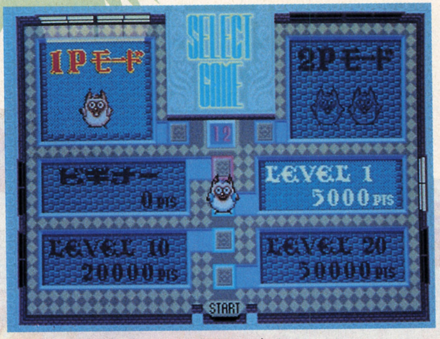
トライアングルパスといえば，なんでタイトルが「キーパー」に変わったのかな？これもJリーグの影響か？

プクルマスターになるのだ

不思議な世界のずっと奥に静かな村がありました。その村のたった1つの泉に，海の化石が埋め込まれた石版が降ってきたのです。こりゃいかん，湧き水を守らなきゃと立ち上がったのがプクルとピクルの2人(匹？）なのだ。

ゲーム画面は泉の上ということらしい。なんで水の上に立ってるの，とは聞かないように。石板だって浮いてることだし。

ゲームが始まると，石板がランダムに出現してくる。これを押したり引いたりして同じ色か，同じ化石の石板を3個以上並べよう。プクルは泉が石板で埋め尽くされないようにひたすらがんばるのさ。



セレクト画面もこんなにキレイ

使うのはジョイスティックと2個のボタン。片方のボタンでジャンプして石板に飛び乗る。そしてもう片方で石板を引っ張るのだ。「レーシーズ」では，石板には乗れなかったし，2個以上の石板をいっぺんには動かせなかった。「キーパー」は自由度が高くなったことで頭を使わなくてもとりあえずは遊べるようになっている。

ときどき外から玉乗り用の玉がころころ流れてくる。メーカーは「木の実」といってるけど，どう見てもこれは玉だい！

これを取ると，光の石板，時の石板といったボーナス石板が出る。光の石板はオールマイティ。どの石版とくっつけてもいい。時の石板はタイムストップのアイテムだ。これらのアイテムに助けられながら，ちょこちょこと石板を消していく。そのうち石板が出るペースが速くなって，並べるのが追いつかなくなりゲームオーバー。

うーむ。わかりやすいけど，どこらへんに刺激があるんだろうか。これが最初の正直な感想だった。

この「キーパー」，これ以上を望まないとこのままで終わっちゃうから要注意。面白いのはこれからなのよ。

もっとうまくプクるには

じゃあ次に，どんなテクニックがあるか考えてみよう。

ルールがわかると最初に思いつくのは，ダブル消し。縦に青なら青，横にヒトデならヒトデを並べていって，交差するところに青のヒトデを押し込む。青の列とヒトデの列がいっぺんに消えて高得点になるんじゃないかなというわけね。

やってみよう。「よいしょ，よいしょ」。ターゲットを決めて配置を作っていくが，この仕掛けが結構場所を取る。使えなくなってしまうスペースが多くて，なんだか自分の首を締めているような気がするな。

よし。赤の巻貝で仕掛け完成。あとは赤の巻貝が出てくるのを待つナリ。

と，緑の巻貝が仕掛けの場所に出現。巻貝の列完成。「Good！」喜ぶプクル。

「ぐーじゃないっ！」

そのまま憤死。

結論。ダブル消しは大変なわりには得るものなし。点数もよくないようだ。

それじゃあ，ほかの消し方のバリエーションを考えてみよう。ただ消すんじゃ面白くない。3個以上並べればいいんだから，4個でも5個でもやろうと思えばできる。やってみると，どうやらこれがキーパーのミソらしい。3個だと100点なのが，4個で1000点，5個並べると2000点入る。もうこれは狙うしかない。

なーるほど，「キーパー」は点数を競う正統派アクションパズルだったのだ。

まず色や形を決め，赤なら赤，巻貝なら巻貝を1列に配置していく。真ん中を残し



X68000用　5″2HD版　8,800円（税別）
サクセス　☎03(3791)2820



ルールはデモでも教えてくれるぞ

て配置できれば仕掛けの出来上がり。あとは「待ち」。石板がくるのをひたすら待つだけだ。プレイが進むとだんだんジャマな石板が増えてきて，こんな配置を作る余裕がなくなってくる。だが，ここがウデの見せどころ。普通の３個並びで満足しているようでは先はない。少しでも点の高い並びを狙うのだ。



狙いは青の石板なのよ



対戦プレイ中。相手はピクル君

もうひとつ。同じ形で同じ色のブロックをまとめて消すと，形だけ，色だけよりも点数がはるかに高い。これも狙いどころ。ベストは同じ色同じ形の石板を５つ並べることなのだが，実際にはちょっとムリなような気がする。なかなか出てこないし，隣に出てきた石板が同じ形だったりして消えやすいのだ。もしできたら何点もらえるんだろう？

ゲームオーバーになると，プクルが登場。「よ，よ」と階段を昇り始める。階段の高さが，自分がレベルいくつまでいったかを表しているというわけ。レベルによってさらにボーナスが入り，自分のランクをAからEまでの５段階で教えてくれる。

これが，ただ「レベルいくつまでいきました」という表示だとたいしてやる気にはならないんだけど，階段を昇るにつれ風景が変わっていくので，つい上が見たくなってしまう。さすがサクセス，演出がうまいんだから，もう。



レベルの階段。目指せランクA！

これで点数表示をディスクに書き込んでくれたら満点だったんだけどな。惜しい。やっぱり点数やタイムを競うゲームは電源切っても記録が残らないとだめだね。

プロの仕事が見事です

キーパーの2Pモードには２種類ある。協力プレイと対戦プレイだ。

協力プレイは1Pと同じルールを２人でやるという感じ。しっかりアイコンタクトをとらないと，高得点は難しいかも。とりあえずプレイは長く続くけど，終盤になると自分が消そうと思ったブロックを相手に消されてしまってヒジョーに悔しい。

対戦プレイは制限時間内に取った点数を競う。お互い，相手が並べようとした石板の配置を崩すという，すごく後ろ向きな対戦になる。どっちもたいした点は取れないレベルの低い争いになるんだな。これだったら，いっそすごーく広いフィールドで５人，10人でバトルロイヤルしてしまったほうが楽しそうだ。まあ，どうやって実現するかという問題はおいといて，だけど。

でも，もとの「レーシーズ」と比べると，さすがサクセス，随所にプロの技を感じる。基本ルールは同じだけど，「レーシーズ」にはプレイヤーを駆り立てるものがあんまりなかったが，「キーパー」には，「高得点の配置を組んで点数を競おう」という意図が明確に出ている。石板の動かし方をより自由にして，ただ並べるだけじゃ面白くないが高得点を狙おうとするとそう簡単にいかないってあたりがうまい。

ゲームに詳しくない人が普通にプレイしても結構遊べるし，点数を狙って挑むとなかなかハードに応えてくれる。幅広い人を相手にセッティングされた，プロの技を感じるパズルゲームだ。

音楽やグラフィックもほのぼのしてファンタジックな雰囲気がよく出ている。なんといってもプクルとピクルがミミズクみたいでかわいくていいやね。「よいしょ，よいしょ」とか「わおー」とかよくしゃべるし。しばらく操作しないでいるとぐーぐー寝ちゃったりして，村の泉が危ないわりにはお気楽ごくらくなヤツらなのだ。

アマチュアのゲームもそれなりにいいと思ったけど，やはりプロの手を経て洗練された姿を見ると，プロの仕事のうまさを感じる(プロにもよるけどね)。両方遊ぶと演出とゲームデザインの重要さがよくわかる。やっぱりプロ化は重要なんだな。……「キーパー」のタイトルはやっぱりそういう意味だったのか？



スコアが記録できればよかったな

うまいぜ！ サクセス

たくさん並べていっぺんに消すのがミソだとわかったら，あとはひたすら反射神経とゲームへの慣れで勝負。石板の配置をパッと見て，ここを押したら３枚揃いそうだなというのを直感的に判断しないといけない。また，プクルのアクションの自由度が高いので，どうするのがいちばん時間がかからず，間違いなく石板を並べられるかというのも考えながらプレイすることになる。

これがマスターできると，次々に出てくる石板をさばくのが快感になる。「冷静に熱くなれる」というのがパズルゲームにはいちばん重要だと思うんだけど，その意味からすると，この「キーパー」，パズルの面白さを全部兼ね備えたナイスなゲームといえそうだ。

総合評価 (0 — 5 — 10)

アクション	★★★★★★★★
グラフィック	★★★★★★★★★
ゲーム性	★★★★★★★★
熱中度	★★★★★★★★

熱く燃える鋼鉄の肉体美



Taki Yasushi
瀧　康史

いきなり現れたロボット格闘ゲーム「マッドストーカー」。格闘ゲームとしてのバランスのよさ，爽快感は保証つきという，なかなか力の入った作品です。対戦モードもサポートしているので，友人と熱い闘いをしてみては？

はっきりいおう。このゲームは面白い。ストリートファイターII（以下ストII）はもうずいぶん遊んでいるけど，ストIIよりずっと面白いかもしれない。え？　ジェノサイド2に似てるって。それは見かけだけだってば。私にとってはロボットモノのアニメがみんなガンダムに見えてしまうように，まだ，ロボットモノのアクションゲームがあんまり出てないだけ。比べてみると見かけ以外はずいぶん違う。

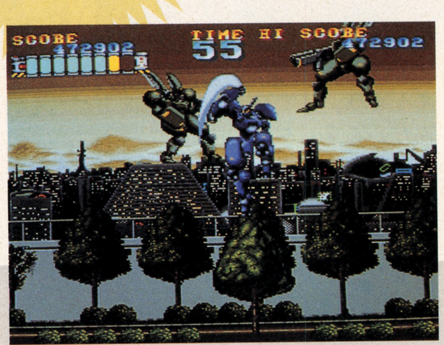
とにかく面白いから買ってみなさい。本当に面白いんだってば。個人的にはいち押しのゲームだと思うぞ。うん。

シナリオモード

このゲームには2種類の顔がある。ひとつはシナリオモード（と勝手に名前をつけた）と，もうひとつは対戦モード。シナリオモードはジェノサイド2に似たゲーム進行。私もジェノサイド2は大好きだったけど，アクションゲームの爽快感はマッドストーカーのほうが上かな？

ステージ数は少ないので，ストレートに進めば全面クリアするまでに30分もかからない。だから，ゲームを解いてそれで終わりというものじゃなくて，ちょっと暇だからやろうかなって感じでプレイができる。

操作は，トリガAが強攻撃，トリガBが弱攻撃。つまり，ストIIライク。ストIIのようにレバー下溜め，レバー上＋トリガAと



昇龍，じゃない大レーザーハウンダーだ

か，レバーを下，右下，右＋トリガA（右に向いているとき）に動かすといった，コマンド系の必殺技がある。

ちょっと違うのは，敵との反対方向にレバーを倒して防御するというシステムではなく，防御は両トリガ同時押しだ。最初は大変かもしれないけど，私は2時間ぐらいで慣れたから大丈夫。それに対戦で遊ぶ場合，防御が完全でないほうがお互いの攻撃が入って逆に面白いかもしれないぞ。

特筆すべきことは，キャンセルがばしばしかかるってこと。つまり，通常の間隔で出る技をキャンセルして，次の攻撃を仕掛けられる連続技を入れることができる。主人公のハウンドドッグで見つかってる最高の連続技は8段攻撃だ。

それにディスクのなかにドキュメントファイルが入っているので，印刷してみれば，連続技，必殺技の一覧がひと目でわかって便利だぞ。

やっぱり対戦だね

ストIIが流行ったように，このゲームが熱いのも対戦があるせいだ。対戦モードでは各ステージのボスキャラや，雑魚キャラの中でも比較的バランスがよいキャラクターが，対戦相手として使える。キャラクターは全部で6体。願わくば，シナリオモードでも6体全部でクリアしたかったんだけどな（続編に期待か？）。

では，各キャラごとに説明してみよう。

●ハウンドドッグ

技の美しさと決まったときの爽快感は，全キャラ中，1，2。オーソドックスな飛び込みからの大キック，大アッパー，大必殺技，そして，飛び込み大キック，小足払い，エルボ×3（キャンセル），ストレート（キャンセル），大オーバーブースト（これで8段攻撃）のような美しい連続技が面白いように決まる。難をいえば連続技の最後が相手に密着するので反撃を食らいやすい。

ハウンドドッグにしかできない2段ジャンプをうまく使い，相手のタイミングを崩して後ろに回り込み，連続技を入れるのが勝利への道だろう。

●ライジングドッグ

連続技の最後が，ライジングアッパークロー（対空技）であることが多いので，相手と一定の間合ができやすく，攻撃が決まったあとの反撃をされにくい。また，この技の出だしが無敵であるため，対空に対しては絶対有利であることを意味する。

代表的な連続技は，飛び込み大キック，小足払い，アッパー，ライジングアッパークローである。個人的にはいちばんかっこいいと思っている。

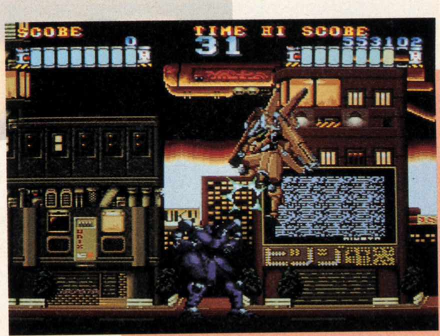
●ゴング

こいつは使ってみるとなかなか面白い。

チャートでは下位のほうに位置しているが，攻撃のバリエーションを考えてみると使って損はない。たとえていえば，ストIIのブランカと，餓狼伝説2の十兵衛を混ぜた



X68000用　5″2HD版2枚組　8,800円（税別）
ファミリーソフト　☎03（3924）5435



この乗っかりが難しいんだよね

ようなキャラクターといえる。

ローリングクラッシュ(ブランカのローリング)は，基本的にダッシュ技のひとつであるが，大と小のスピードの違いが大きいので，敵のタイミングを崩すのにも使える。ほとんど地面に足がついたのが見えないくらいの速さで，次のローリングが出せるため，小を出して敵の返し技(たとえば立ち大技など)をかわして，敵の返し技が戻るモーションのときに攻撃することができる。

ブーメランクラッシュ(バーチカルローリング)では，相手をめくることができるのも覚えておこう。大はかなりの高さまで上がり，小はすぐに落ちてくるので，連続して出されると，キャラクターによってはものすごく辛いものがある。

1カ所に停まることなく，相手を翻弄させながら，たまに真下に溜めておき，ダッシュ1本背負いを出すのがよい。さらに，ダッシュ1本背負いは小で出すと，出だしの少しの間無敵なため，敵の空からの攻撃をかわしてさらに投げることができるし，大は逆に技が出る徴候がまったくないので連続技にもなる。

●シルフィード

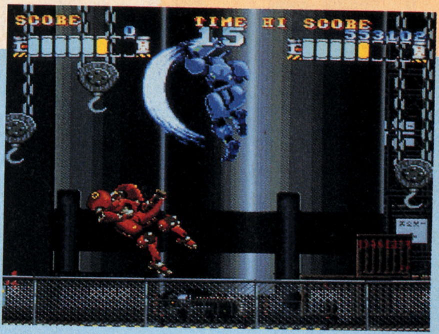
いちばん癖があるのはこのキャラクターだろう。

制空権は完全にコイツが握ってると考えてよい。なぜなら空中を飛行できるのがシルフィードだけだからだ。普通のジャンプでさえ軌道というものがなく，自分のスティックさばきで着地点を自由に変えられる。そのためただのジャンプ大キックでさえも最初はうまくいかない(垂直ジャンプから横にずれるべし)。

また，必殺技を使えば，スティックさばきでいくらでも着地点を変えられるが，相手の頭の上に着地することが難しいほど左右に動くのが速い。使いこなせば，ハウンドドッグのスプレッドショットを飛び越え，ハウンドドッグが硬直中に，相手の頭に乗っかることもできる(シナリオモードで，難易度ハードにすれば，コンピュータが嫌というほどやってくれる)。



見よ！　紫電断烈斬のこの迫力



対戦では最強最悪のボスもこのとおり使える

連続技の基本は，立ち大キック(キャンセル)，大パイルバンカー(3段)で，これに飛び込みを加えると4段になる。

敵をうまく翻弄させ，パイルバンカー主体の攻撃にときおりスライディングを交ぜるのがよいだろう。

●プリソナーβ

必殺技のグランドバキューム(相手を吸い込んで抱き締める)の吸い込み中は上半身無敵なので，スプレットショットが当たらない。吸い込んだあとはレバーを左右に振ったりして延長することができる。

敵を放したあとは，ダイナマイトチョップを放っておこう。相手が油断してればダメージを与えられるし，防御されていてもヒットポイントを削れる。そして再び吸い込むこともできる。このへんは読み合い(ちょっとダークかな？)。

戦法としては，敵が技を出した直後の硬直を狙って吸い込むことで勝負する。

●神威

とにかく攻撃能力が高い。飛び込み大斬り，大斬り(唐竹割り)，紫電断烈斬で5段攻撃という技がある。通常技も含めてすべての攻撃力が異様なまでに強いので，ゲージは2マスも(16ダメージ)も減らせるのだ。もし，連続技の前になんらかの攻撃を与えれば気絶してしまうので，もう一度連続技を決めることもできる。

居合突きとはりせん斬りは，ほとんど飛び道具といっていいくらいリーチが長くダメージ量も多い。このキャラクターには飛び道具がないが，相手が飛び道具を使ったときに相討ち覚悟ではりせん斬りをすれば，ダメージ量でもとが取れてしまう。また，つむじ討ち(斬影拳)は小なら移動中が無敵なので，飛び道具をかわして攻撃することができる。

対空技は，唐竹割りぐらいなのがこのキャラのネックといえる。どちらかといえば，一撃必殺キャラなので，隙を見せると投げられてしまうキャラクター(ゴングとか，プリソナーβなど)には苦戦する。

弱肉強食対戦チャート

表1の数字は私が友人数人と，ひとつのキャラクターを極めるまで実際にプレイをし，「人間同士」の対戦での有利，不利を表しているものだ。横軸のキャラが，縦軸のキャラと対戦したときに5分5分であれば5，多少有利に戦えるならば6，キャラによる差が明らかに現れるようなら7という基準でチャートを作ってみた。

表1を見てもらえればおわかりになるだろうが，ストⅡやほかの対戦ゲームに比べて，びっくりするほどバランスが取れている。これは，8-2，7-3などの組み合わせが出てこないことでもわかるだろう。

ひと言でいってしまえば，これはすごいことだ。いかにゲーム開発者が，このバランスのよさを求めるために力を注いでいるかわかるだろう。

表1　弱肉強食チャート

	ラ	ハ	神	シ	プ	ゴ	T.t
ライジングドッグ	＼	5	5	6	6	6	28
ハウンドドッグ	5	＼	4	5	6	6	26
神威	5	6	＼	5	4	5	25
シルフィード	4	5	5	＼	5	5	24
プリソナーβ	4	4	6	5	＼	5	24
ゴング	4	4	5	5	5	＼	23

チャート作成協力：斉藤　昭夫

不満点も少しはあるかな？

出来があまりにもよいせいで，不満な点がいくつか残る。シナリオに矛盾ができてしまうからやってないのだろうが，やっぱりすべてのキャラクターでクリアしたい。ぜひとも，プリソナーβでクリアしたいのだ。

それから，よくある格闘対戦もののように，敵のボスとだけ戦うモード，つまりVS CPUモードも欲しかった。

あと，最初のメニュー画面で，選択できるのは1プレイヤーのみなんだよな。やっぱり再度対戦するか否かは，負けたほうが決めなくっちゃー熱さにかけるぜ。そうそう，音源ドライバはZ-MUSICだったりする。

総合評価　(0　5　10)

項目	評価
印象	★★★★★★★
ハマリ度	★★★★★★★★★
連続技	★★★★★★★★★
続編が欲しい度	★★★★★★★★★
音楽	★★★★★★
格闘スピリッツ	★★★★★★★★★
キャラクターでかい度	★★★★★★★

新たなる闘いのはじまり



友人を呼んで対戦するもよし，コンピュータ相手に美しいプレイを追及するもよし。寒さを吹き飛ばすくらい，思いっきり「餓狼伝説２」で熱くなろう。でも，熱くなりすぎて，友達をなくさないようにね。

Asakura Yuji
朝倉 祐二

待ちに待った製品版が送られてきました。先月号に続いて「餓狼伝説２」のレビューをお届けします。

製品版の内容

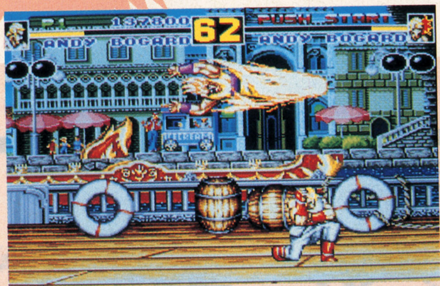
最初に先月紹介できなかったおまけのジョイパッドですが，使い勝手は良好です。大型で厚みがあり曲面構成のボディは手に馴染みやすいデザインでしょう。ボタンはスーパーファミコンのようにL，Rボタンがついているタイプです（全部で６つボタン）。このパッドがついて9,800円は割安感があります。

ゲームスピードは10MHz機で遊ぶとたまに動きが重くなりますが，16MHz以上のマシンだと快適に遊べます。処理速度を少しでも軽くするため，画面上に置かれている樽などの表示をしないように設定できるので，各自対応するといいでしょう。

また，フロッピーディスクで遊ぶと対戦相手，対戦場所が変わるたびにディスク交換が頻繁にありますし，データの読み取り時間も短くありません。できることならハードディスクにインストールして遊びたいゲームです。BGMは内蔵FM音源のほかに，MIDI（SC-55，MT-32）に対応しています。ただし，しっかり移植された映像関係に比べると，BGMの移植はややレベルが低いかな，という気がしました。



X68000用　5"2HD６枚組9,800円（税別）
魔法株式会社　☎078(261)2790



いまいち使えないアンディの超爆破弾

キャラクター紹介

●テリー・ボガード

「餓狼伝説２」の主人公キャラクター。必殺技も多彩で，なかでもライジングタックルが強力です。出始めが無敵状態となるために，起き上がりに出すと敵の攻撃に当たらず攻撃することができます。ほかに大キック（キャンセル）パワーウェーブ，パワーウェーブ→クラックシュート→投げなどの連続攻撃があります。決して強いキャラクターじゃないのに，なぜか人気のあるキャラクターです。

●アンディ・ボガード

テリー・ボガードの弟。「餓狼伝説２」最強のキャラクターといわれています。特に斬影拳が強すぎます。斬影拳をガードしても，すぐにしゃがみ大キックを出されると，たいてい食らってしまうのです（いわゆるハメ技）。斬影拳の連打はほどほどに，小パンチキャンセル昇龍弾くらいにしておきましょう。

●東　丈

ムエタイの使い手。必殺技の種類は多いです。タイガーキックは出始めが無敵なのでかなり使えますが，ゲーム中だと「タイガーキック」のボイスが「タイヤキーッ」に聞こえます。勝ちポーズの「よっしゃぁー」は本当に気持ちよさそうです。

●ビッグ・ベア

レスラー。図体もでかく動きも鈍いので，使っている人をあまり見たことがありません。投げ技が多彩なのとジャイアントボムの出始めが無敵ですから，うまく使えば強いと思います。しかし，四天王相手には苦しいかな。

●山田　十兵衛

柔道の使い手。70歳を越えているのに，いまだ現役の女好き爺さん。スライディングになる大キックとダッシュ２本背負いが強い。相手が転んだら，すかさずダッシュ２本背負いをかけましょう。「わしの勝ちじゃー」と叫びながら繰り出す超必殺技はプレイをしていて快感になります。



お年寄りはいたわろう



うぐぉあ！　こりゃ強烈に痛そうだ



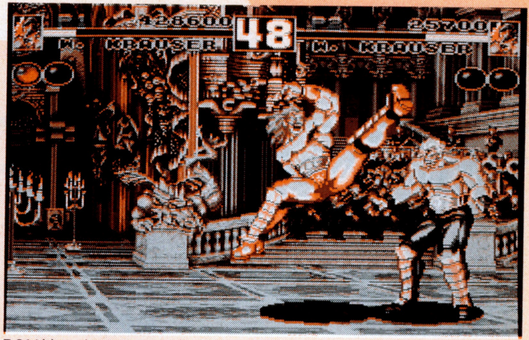
超必殺忍蜂が炸裂！

●チン・シンザン

破岩激と気雷砲が使えましゅ。チン使いには強い人が多いようでしゅ。イカサマ商人のような風貌をしているくせに，幼児言葉を使うのはやめてもらいたいでしゅ（読みにくいけど，実際こうなんだもん）。やられたときの「あいたっおっほーほー」はとても情けないものがありましゅ。



同キャラ対戦は実力が勝負を決める



BGMはレクイエム。安らかに眠れ

●キム・カッファン

テコンドーの達人。半月斬を初めとして飛燕斬，飛翔脚とすべての必殺技が実用的です。セコイやり方ですが大パンチ，大キック同時押しによるライン飛ばし攻撃をガードさせ，キャンセル半月斬を繰り出すことによって相手の体力をジワジワと削っていくこともできます。しかし超必殺技あってのキム，体力ゲージが赤くなったら迷わず鳳凰脚を狙いたいものです。相手にガードされるとなにもできないのが欠点ですが，相手と密着した状態で出せば必ず入ります。敵の攻撃をわざと食らって鳳凰脚を出す「当て身鳳凰脚」も使えるでしょう。

●不知火　舞

くノ一，つまり女忍者。露出度の大きいコスチュームといい，勝ちポーズの「にっぽんいちぃ～，ぷるるん」といい，舞には羞恥心というものがないようです。動きは素早いのですが，いまいち使える必殺技がありません。龍炎舞は「ニューウェーブ」に聞こえるし，華蝶扇は「課長さん」に聞こえます。そんな舞ですが通常技のしゃがみ大キックのリーチの長さと，ジャンプ小キックは，かなり使えます。また，超必殺技は全登場キャラクタのうち，もっとも出しやすく，かつガードの上からもガンガン体力を削っていくので，体力ゲージが赤くなった舞は要注意。女は怒ると怖いですからね。

対COM戦における戦法

少しだけ闘いを有利に進めるための戦法



これが噂の日本一～プルルン

を伝授します。対COM戦の場合に絶対にマスターしたいのがライン移動攻撃です。

自分とCOMが同じラインにいるとき，こちらからライン移動（小パンチ，小キックボタン同時押し）します。COMがあとを追っ掛けてきたら，ガードしてもいいし，もう一度ライン移動をしてもかまいません。とにかくCOMと別のラインに移動するのです。そうしてからレバーニュートラルの位置で，パンチボタンを押します。

すると，自動的にCOM目掛けてマイキャラがライン移動攻撃してくれるのです。キックボタンだと移動の軌道が大きくなってしまい相手に反撃のチャンスを与えますから，絶対にパンチボタンを使ってください。そして，ライン移動し始めたら，相手の反撃に備えてレバーを防御方向へ倒しておきます。ここでコンピュータの取ってくる行動は２つあります。

1)　ライン移動攻撃をガード

COMがライン移動攻撃をガードしたらすかさずCOMに歩み寄って投げます。

2)　避け攻撃をする

避け攻撃とは敵が攻撃を仕掛けてきたときに，レバー→＋トリガAで繰り出す攻撃のこと。で，文字どおり相手の攻撃を避けて攻撃してくるために，これをCOMにやられるとダメージを与えるどころか，こちらが逆にダメージを受けてしまいます。しかし，ライン移動中にレバーをちゃんと防御方向に倒しておけば攻撃を受けることはありません。避け攻撃をガードしたら，大キックをお見舞いしてやりましょう。

以上の技は対戦相手がビッグベア以外のキャラクタのとき使うことができます。このライン移動攻撃からの投げは，いわゆるハメ技なので，対人間で多用すると嫌われものになること間違いありません。

ほかにも対COM戦の特定キャラクタに対する有効な攻撃技はたくさんありますが，自分で探してみましょう。

BGMがよければ

結論として，NEO・GEO版を持っているユーザーから見ると不満点は多いでしょう。実際にオリジナルと遊び比べてみましたが，キャラクターの動きがいいぶん，BGM特にSEが貧弱な点で損をしています。そして，前回ほめていたグラフィックも，やはりオリジナルのほうが上という感じですし（決して見劣りするわけではない）。

しかしX68000という拡大・縮小をもたないハードウェアで，NEO・GEO版に迫る作品を作り上げたことは評価できますし，いやがうえにも次回作に期待が高まります。シリーズの流れでいくと「餓狼伝説SPECIAL」ですが，個人的には「サムライスピリッツ」がいいですね。そのときにはBGM，SEの移植にも力を入れてもらいたいものです。

では，皆さんも次回作への期待を胸に秘めつつ「餓狼伝説２」を遊びましょう。

ゲームが主役ならBGMはわき役か？

「アウトラン」以来，ゲームにとってBGMはただ鳴らせばいいというものではなくなりました。ゲームの世界でも映画のように映像に合った音作りが重要視されるようになり，いまやゲームと音楽は別々のものでなく，ひとつのものと考えるのが普通です。BGMを気に入ってゲームを買う人は世の中にはたくさんいますし，かくいう私も「餓狼伝説２」ではテリー・ボガードのステージのノリのいいBGMが大好きで，それだけでNEO・GEOを買ったようなもんです。

そういう理由もありX68000版の「餓狼伝説２」は動きだけでなく，BGMにも期待していました。しかし，思い入れが強かっただけにMIDIで聞いたときは残念な気分になりました。BGMがゲームの印象を変えてしまうことを教えてくれた１本です。

総合評価（0　5　10）

グラフィック	★★★★★★★★
操作性	★★★★★★★★
BGM	★★★★★★
オマケパッド	★★★★★★★★

初心者のための1コインクリア入門



Nakano Shuichi
中野 修一

ストII'の簡単な攻略をまとめてみた。12人すべてを究めるのが目標だ。なお，攻略の半分以上は対人戦にはほとんどきかないので注意。え，簡略すぎてわからない？　最大の秘訣は「負けないように戦うこと」だそうだ。

アーケードゲーム史上で記録的なヒット作となったストII'がついに移植された。「話題のゲームだからして一応買ってはみたけど，必殺技は出ないし，敵は強くて，なんじゃこりゃ」状態の人もいるのではないかと思う。

確かにストIIシリーズは対戦が基本のゲームではあると思う。しかし，1人用でもできるということはやはり重要なことだと思うし，できるようになっている以上は極められねばならない。ということで，対コンピュータ戦（以下対C戦）の攻略をやってみようと思う。

ところで，ストII'は6ボタンを前提にしたゲームだ。すでに多くの人がCPSファイターやメガドラの6ボタンスティックを使ってプレイしている，と思う。

しかし，X68000版はこういったスティックなしでもプレイできるようになっている。2ボタンスティックでもちゃんとできるということになっている以上，これも極められねばならない。ということで2ボタンスティック対応の攻略である。

ちなみにメニューにはキーボードでも操作できるように書いてあるが，これはジョイスティックと併用して使えということなのだろう（と，理解している）。ザンギエフが簡単にグルグルできるようにという配慮に違いない。



目標は全キャラノーミスクリアだ

基本操作について

初心者用ということで，あまり難しい技は要求したくないのだが，最低限，立ち防御，波動拳，投げの3つができるようになっておいてもらいたい。特に対戦であれ対C戦であれ，瞬時に立ち防御ができることが必要とされる。くれぐれも跳び上がったりしないように。練習法は立ち弱キックとしゃがみ弱キックを高速に繰り返すというのがいいだろう。

もうひとつ注意。ストII関係は専門用語が多い。「サイコクラッシャーアタックはベアアタックで落とせ」とかはまだしも「低蹴打で牽制しつつブーメランフックを顎狙突拳で迎撃，寄ってきたら元伝暗殺蹴で……」となると初心者にはチンプンカンプンだろうし，「サマーソルトスカルダイバーでくるのでフライングメイヤーを狙わないほうがいい」くらいになるとわかる人のほうが珍しい。ここでは雰囲気でわかりそうな俗称を使用することにした。

では，コマンド系の技のうち，昇龍拳なし，スクリューパイルなし，ヨガフレイムなしという条件で全キャラクリアのための攻略をしてみたいと思う。ランクは常に最高に上げておくこと。敵の情けは受けるな。

はっきりいってコンピュータはずるい。鉄則は「一度試して駄目だった技は二度と使うな」と「一度試して有効だった技はどんどん使え」ということになる。多分に美しくないプレイになるが気にしないように。

スペースの都合で四天王はおいといて，基本8キャラについて解説する。

リュウ/ケン 6・2・3とパンチせよ

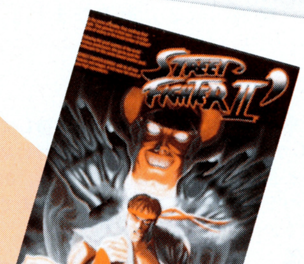
初心者向け基本キャラ。波動拳はしゃがみ位置からレバーを前に押し出すようにパンチだ。いずれは昇龍拳を使いこなしてほしいものだ。設定は大パンチ＋中キックが妥当か？（中パンチでも可）

vsリュウ

単純なのが「波動拳で打ち勝つ」こと。とりあえず波動拳の練習ステージのつもりでやってみる。波動拳は2つセットでくる確率が高いので跳び込んで上下に決める。基本中の基本。相手は堅いが守れば勝てる。

vs本田

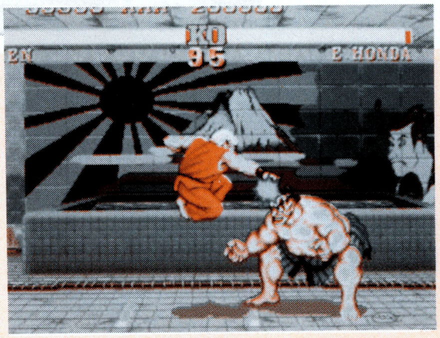
跳び込みパンチを練習しよう。かなり離れたところでも当たる。絶対に有利な間合いというものを体で覚えること。



X68000用5"2HD 4枚組 12,800円（税別）
カプコン

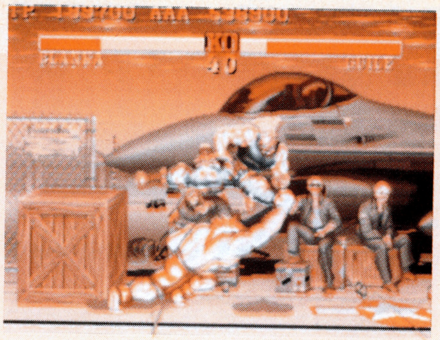


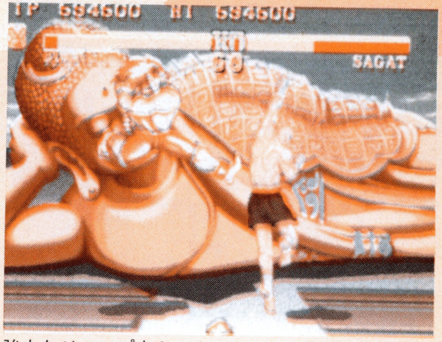
跳び込みパンチはケンのほうが強力



余裕があれば昇龍拳の練習


対空兵器として使えなくもないが……


パンチに続いてローリング

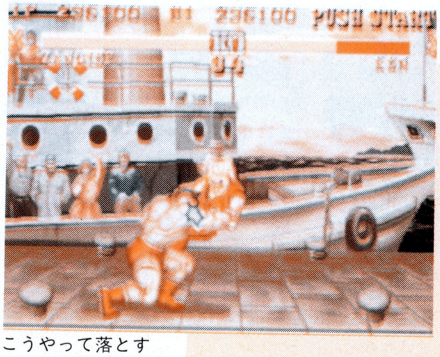


強力なジャンプ中キック

vsブランカ
とりあえず，鼻先まで引き付けて波動拳と離れて波動拳。半端な間合いは命取り。跳んできたら引き付けてアッパー＋波動拳。
vsガイル
離れたら波動拳（強はあまり意味がないが)。跳んできたら確実に落とす。立ち強パンチがよい。足払いのあとは波動拳。
vsケン
離れた位置から跳び込んできたら着地点にあわせて波動拳。頭の上はアッパー。
vs春麗
どうやっても負けそうにないが，跳び込んできたら引き付けてアッパー＋波動拳。昇龍拳の練習には最適。
vsザンギエフ
三撃離脱を繰り返すか，離して波動拳と足払いが基本。寄ってきたら鼻先で波動拳。リュウなら竜巻旋風脚だけという手もある。
vsダルシム
無闇に跳び込まない。
vsバイソン
しゃがみ弱パンチ連射……というのは6ボタン向けだからおいといて，無難なのはアッパーの空振りに足払い。跳んできたらしゃがみアッパー。２本目はしゃがんで大パンチ以外のボタンを連射するだけ。
vsバルログ
転ばせて跳び込んで蹴る。バック転を追いかけて投げる。バルセロナアタックは逃げ蹴り（後方ジャンプキック）かその場でタイミングよく立ち大パンチ。
vsサガット
離れた間合いでジャンプパンチ（またはキック）で跳び込む。タイガーアッパーカット封じを身に付けること。まあ，無茶をしなければ負けないと思うが……。
vsベガ
倒れた相手に跳び込むとき以外斜めジャンプ厳禁。ジャンプはすべて垂直ジャンプで。踏みつけにくる相手より先にジャンプして落とす。飛び込んでくるときはひたすら防御。寄ってくるときは足払いで耐える。

ブランカ 爪がうなるぞ歯には歯を

投げがないのがつらいが，パワフルで使いやすい。設定は強パンチと中キックだ。
vsリュウ
波動拳を跳び越えて攻撃。空中でも優位。
vs本田
中パンチに設定している場合は３段攻撃の練習ステージ。大パンチならガンガン攻めるのもよし。基本は相手が浮いたらローリング。足払いは厳禁。スーパー頭突きは垂直ジャンプ中キックで落とす。
vsブランカ
跳び込みはローリングで落とす。ローリングはジャンプキックで返す。
vsガイル
相手が跳ぶか大足払いでローリング，基本はこれだけ。手早くやりたいときは跳び込んでくる相手をのびーるパンチで落とし，直後にローリング。これは手前で止まるので当たらないが，敵はサマーソルトを空振りするのでこれにローリングを当てる。
vsケン
相手がジャンプした瞬間にローリング。空振りの昇龍拳は確実に落とす。
vs春麗
跳んでくるのでスカートめくり（立ち強パンチ)。または飛んでくる背中に当てるようにローリング。着地点でできれば電撃。
vsザンギエフ
垂直ジャンプキックのみが安全だが，引き付けてローリングのみでも可。



こうやって落とす

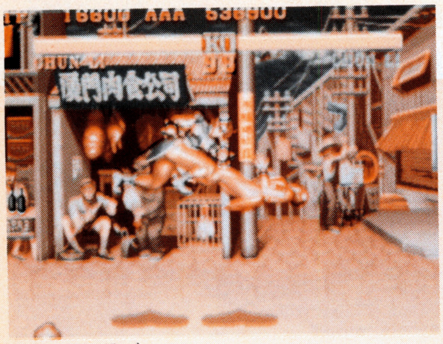
vsダルシム
ジャンプ中キックで押す。頭の上をうろうろされたら着地点で中足払いひとつ。
vsバイソン
電撃だけ……なんだが，強電撃はきついか？　２本目はのびーるパンチ連打が早い。
vsバルログ
三角跳び蹴りは垂直ジャンプキック。バルセロナアタックをローリングで返す癖はつけないほうがいい。
vsサガット
上タイガーをかわしてローリング。ジャンプ中キックはアッパーカットも抑える。
vsベガ
楽勝。開始直後にローリング，倒れた相手が立ち上がったら１拍おいてローリング。宙に浮いたベガはすべてローリングで落とせる。中足払いは絶対に負けない。

ザンギエフ 7つの投げを持つ男

スクリューなしでなにが面白いんだ？という気もするが，破壊力があり，意外に楽に進める。設定は中パンチ＋強キック。
vsリュウ
波動拳を跳び越え蹴る。
vs本田
アドリブで。パワーで押せる。
vsブランカ
足払い。寄ってきたら隙を見て投げる。
vsガイル
ソニックは完全によけよう。跳んできたら立ちキック。



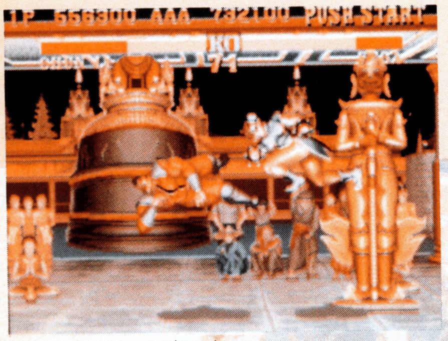
攻めまくるのも手だ



開始直後の投げ



百裂をまじえた連続技は強力



頭上に浮いたら必ず投げる

vsケン

近めのジャンプキックはしゃがみパンチで落とす。

vs春麗

三角跳びできたら足払い。つらいねぇ。

vsザンギエフ

垂直ジャンプキック＋着地足払い。最初の蹴りをなるべく低い位置に当てること。足払いは必ず出すこと。

vsダルシム

寄ってきたら投げるか足払い。

vsバイソン

上に同じ。一度転がして蹴りまくりも可。

vsバルログ

三角跳びは立ちキック。

vsサガット

跳び込んで蹴る。パワーで押せる。

vsベガ

寄ったら足払い。跳んだら立ちキック。画面端でジャンプするとサイコで突っ込んでくるが，落下速度が遅いので落ち際に蹴るとちょうど当たってくれる。

春麗 私の靴をおなめ！

殴る蹴る投げる踏むと4拍子揃ったキャラ。動きが速くて扱いやすく，足が太い。設定は中パンチ＋中キック。

vsリュウ

結構きつい。地道に飛び込み上下2つ決めていく。昇龍拳の空振りは確実に投げる。

vs本田

歩きながら中パンチで押す。跳び込みは中キックで落とす。スーパー頭突きはスピニングバードキックで返す。

vsブランカ

くぐり投げを覚えること。確実なのは足払いを当ててスピニングバード＋投げ。

vsガイル

跳んでくるのを落とす。近づいてきたら足払いを防御させてキックで跳び越し，着地と同時に投げる。ガイルの跳び込み技はしゃがんでいればほぼ当たらない。

vsケン

手は出さず，昇龍拳の空振りを投げる。

vs春麗

くぐり投げと空中投げの練習ステージ。跳んでいる相手の着地点よりやや内側へ移動し投げる。三角跳びの後ろから投げる。

vsザンギエフ

スピニングバードキックのみ。連発ができるようになろう。

vsダルシム

ひたすら踏む。頭突きをかわして投げる。

vsバイソン

跳んでくるのは立ち中パンチで落とす。あとは足払い。2本目はしゃがみ中パンチ。

vsバルログ

スピニングバードがなぜか当たる。

vsサガット

踏んで投げる，という手もあるが無難に待ちで空振りアッパーを空中投げしよう。

vsベガ

真上に上がれば空中投げ，足払いをガードさせてすぐスピニングバードあるいは跳び込みキック＋踏みつけ。サイコは踏める。

ダルシム いつも心にヨガファイヤー

扱いの難しさはピカ1。連続技もなく苦しい。使いこなせば強い……のか？　設定は強パンチ＋中キック。

vsリュウ

最大の敵。コツはヨガファイヤーで撃ち負けないこと。波動拳をよけること。絶対に寄せないこと。跳んできたら中キック，ジャンプパンチ。

vs本田

ヨガファイヤー＋跳んできたら中キック（以下ヨガ足と略）。

vsブランカ

基本はヨガ足。近づいてきたら，しゃがみパンチで跳ばせる。寄せないこと。

vsガイル

ヨガ足。中スラ連発という手もある。

vsケン

ヨガ足。頭の上を越えさせないこと。

vs春麗

ひたすら立ち中キック連打のみ。

vsザンギエフ

最初に引き付けて燃やし，あとはヨガ足。完全な操作が要求される。寄ってきたら逃げながら投げる。

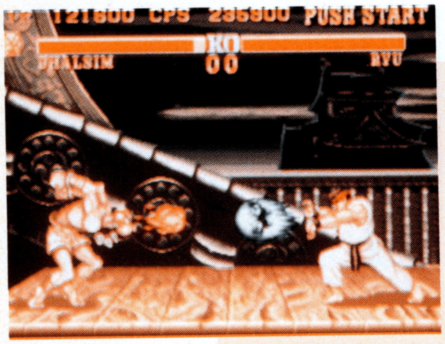
vsダルシム

アドリブで。中スラのみという手もある。

vsバイソン

ひたすらヨガ足。

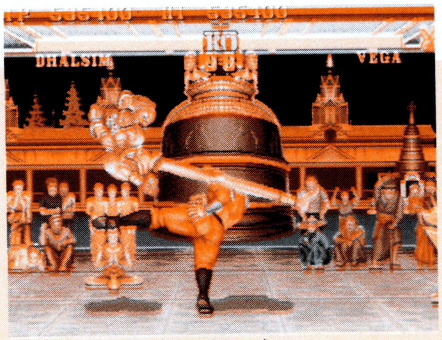
vsバルログ



負けるなヨガファイヤー



逃げながら投げる



寄ってきたら顔を狙ってキック

ひたすら中スラ。

vsサガット

アドリブでなんとかする。

vsベガ

最難関。近寄ってきたら垂直ジャンプ中キック。ヘッドプレスで近くに着地したらすかさず投げる。無駄な動きは厳禁。

ガイル 戦いのプロフェッショナル

基本的に強い。対空にも優れ，投げも強い。設定は大パンチ＋中キック。または中パンチ＋中キック。

vsリュウ

波動拳は先読みで跳び込む。接近時はソニックを波動拳で消させ，直後に裏拳。

vs本田

足払い＋サマーソルトの基本形。

vsブランカ

離れてソニック。跳んできたらサマーソルトまたはくぐり投げ。

vsガイル

ヤな奴。無駄な動きをなくすこと。

vsケン

跳んできたら落とす。

vs春麗

サマーソルトを相討ちにされないタイミングをつかむこと。

vsザンギエフ

足払い＋跳んだらサマーソルトの典型。

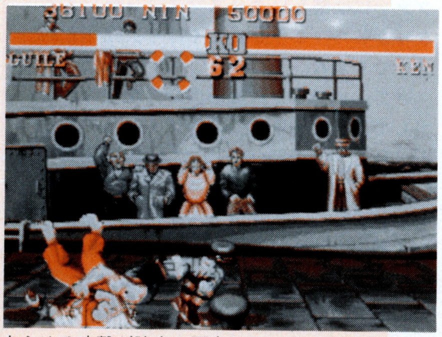
vsダルシム

アドリブで。

vsバイソン



跳んできたらサマーソルトで落とせ



あらゆる攻撃が強力。死角はない？



腹でつぶせ

足払い＋サマーソルト。

vsバルログ

サマーソルトを相打ちにされないように。バック転は止まった瞬間に投げる。

vsサガット

アドリブで。

vsベガ

空中投げは確実に。飛び込みはサマーソルトで落とさずアッパーで。

本田 日本の国技は最強だ

飛び道具に弱いがスーパー頭突きと張り手は強力。設定は中パンチ＋中キック。

vsリュウ

波動拳を跳び込んで腹。間合いが命か。

vs本田

百裂張り手のみ。

vsブランカ

跳んできたら頭突き。隙を見て投げる。

vsガイル

跳んだら頭突き。へたに足を出さない。

vsケン

危ないことはしないように。

vs春麗

跳んできたら引き付けて頭突き。スピニングバードはしゃがみパンチ。

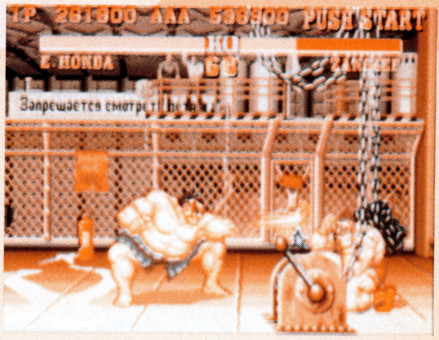
vsザンギエフ

百裂張り手のみ。または頭突きのみ。

vsダルシム

スライディングは腹でつぶし，足払い。

vsバイソン



とりあえず百裂張り手

跳んだら頭突き。適当にあしらう。

vsバルログ

跳んできたら垂直キック（頭突きも可）。

vsサガット

下タイガーは頭突きで返す。

vsベガ

跳んできたら中頭突き。寄ってきたら中頭突き。頭上に跳んだら腹で落とす。

まとめ

煽り文句の「すべての技を駆使して相手を倒せ」というのは，すべての技を駆使しないと相手は倒せないということだ。2ボタンではちょっときついところもある。ま，スーパーストIIの対C戦に比べればマシかもしれないが（HARDESTモードがいかんのかなあ）。

ストIIは初めてという方もぜひ，今回の攻略をもとにノーコンティニュークリア，ノーミスクリアを目指してみてほしい。

アーケード版でこの作品が出た頃は日本一決定戦などの話題だけでも盛り上がったものだ。海外での人気も高く世界大会という噂まで出てきた。きっと，全キャラを駆使し対戦無敗を誇るドイツの天才少年とか，どんな体勢からでも7つのキャンセル技を繰り出す香港の無敵リュウ使いとか，ヨガカメレオンのまま相手を翻弄するウクライナの超能力少女とかが現れるのではないかとワクワクしていたのだが……，あの話はどうなったのだろうか？

あとは対戦あるのみか？

微妙な反応やキャラのアルゴリズムなどはほぼそのまま移植されているのがうれしい。難易度設定はなぜか違っているみたいだが……。

「ゲームってのは音だねぇ」というのは昔書いたが，音楽は少しだけ違う気がする。内蔵音源版はまだましか？　CompactXVIの内蔵スピーカを前提にしたのではないかと思われるようなイコライジング設定なのでオーディオ機器につないで聞くようなことはやめたほうがいい。あるいは低音を1/3くらいにして聞く。

効果音がぺしぺしなのはちょっと悲しい。ボリュームを絞ってほとんど音なしでやると結構いいぞ。

まあ，音以外は「よくぞここまで……」というデキ。10MHz機だと動きがスーパーストIIになってしまうが，10MHz＋ADPCM4.Xでも遊べなくはない速度だからいいか？

総合評価	0　　5　　10
グラフィック	★★★★★★★★★
音楽	★★★★★★
効果音	★★★★
移植度	★★★★★★★★★

TREND ANALYSIS



1994年1月号のハガキ集計ベスト10

最近買って気に入ったソフトは？

POINT	タイトル	発売元	発売日
278	ストリートファイターIIダッシュ	カプコン	'93/11/26
56	ドラゴンバスター	電波新聞社	'93/12/10
42	悪魔城ドラキュラ	コナミ	'93/7/23
33	ぶたさん	電波新聞社	'93/10/29
19	餓狼伝説2	魔法株式会社	'93/12/23
17	ネメシス'90改	SPS	'93/11/12
16	MATIER Ver2.0	サンワード	'93/10/20
16	コットン	EAビクター	'93/9/24
9	SX-WINDOW ver.3.0	シャープ	'93/3/30
9	X68000傑作ゲーム選	アスキー	'93/11/12
8	クレイジークライマー/クレイジークライマー2	電波新聞社	'93/8/27

（無作為抽出した1000通のハガキを集計）

新記録達成！　1位のポイントを見てください。10月号「悪魔城ドラキュラ」のポイント数243がとうとう破られました。

発売決定前から，移植希望の声が非常に高かった「ストリートファイターIIダッシュ」ですが，みんなの期待に見事に応えてくれる出来でした。移植の噂がささやかれ始めてからもう1年以上。前作「ストライダー飛竜」の発売がちょうど1年前の11月27日でしたから，カプコン，満を持しての登場です。いやがおうでも高まる期待に胸ふくらませ，理想のものを手に入れられるかどうかの不安におののいていたファンの皆さんの喜びははかり知れません。やはりプレイは6つボタン，とCPSファイターの人気も高いようで，しかも2つ購入という人が続出との話。別売のコネクタ4,000円と合わせると出費もなかなかですが，それだけ価値ある移植作品だったということでしょう。今後の票の獲得状況が気になります。ただ，熱中するあまりか，推薦理由には「いうまでもない」「当然」といったようなものが多いのは編集部としてはちょっと寂しいところです。プレイした感想などもお待ちしていますね。

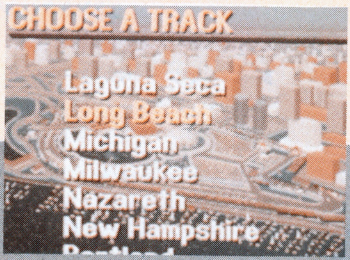
で，3カ月間(12月号では集計がお休みだったので実質は4カ月)ダントツだった1位の座を明け渡してしまった「悪魔城ドラキュラ」は，3位に転落。2位に入ったのは電波新聞社の「ドラゴンバスター」です。根強い人気のビデオゲームアンソロジーシリーズの第7作め。「ナムコの先見の明が感じられる」との声もあるように，当時としては新鮮なシステムで，その後のゲームへの影響の大きさがうかがえます。しかし，単なる懐かしさではなく，いまプレイしても面白いということが獲得票に結びついているのは，もちろんです。

5位以下は完全に票が割れてしまいました。そのなかでも注目されるのが魔法株式会社の「餓狼伝説2」。前作「餓狼伝説」からわずか5カ月のスピード移植ですが，当初の予定よりも発売を伸ばして内容に検討を加えたとのこと。X68000版オリジナルの四天王対戦や4つボタンパッドの同梱など，ファンには嬉しいクリスマスプレゼントとなりました。こうなるとやはり次回作も期待，ですよね。

さて，1993年は粒揃いの新作が次々と登場してユーザーを楽しませてくれた年でした。今月号のアンケートはがきはGAME OF THE YEARの投票用紙になっています。各項目の狭いスペースだけではあき足らない人や，自由応募部門などには，官製はがき，封書なども受けつけていますので，熱いご意見を送ってくださいね。

ウワサのソフトウェア(海外編)

INDY CAR RACING







PC互換機は，しばらく前から圧倒的なCPUパワーを背景にしてAMIGAに代わり3Dシミュレータ界の盟主の座についている。私は最近までは「AMIGAでもまだまだいける，結局勝つのはセンスのよいシステムデザインを持ったマシンだ」と信じていたのであるが，今度ばかりはその信念も崩れ落ちた。

PC互換機用のINDY CARレーシングシミュレータ「INDY CAR RACING」である。制作はあの「Indianapolis 500: the Simulation」(以下Indy 500)のPAPYRUS。ダイナブックやAMIGA500であれだけ動くソフトを作った連中に486マシンを与えたらこういうものを作るという見本のようなシミュレータである。

＊　＊　＊

とにかく度肝を抜かれるのがそのグラフィック(といっても最近のPC互換機では珍しくもないが)。リアルタイムのテクスチャマッピングにより，実在のインディカーの外観が完璧に再現されている。すべてのドライバーが車体とヘルメットで識別できるといえば，それがどんなレベルのものか想像つくだろうか。車体，路面，壁，どこをとってもテクスチャマップで，およそベタ塗りの面など見あたらない。それでいて完全リアルタイム。さすがに66MHzの486マシンくらいはほしいところがあるが，グラフィックのディテールを最高にしてもストレスなく遊べる。３Ｄの利点を生かしきって，デモではダイナミックな視点変更で見せる。

音響関係も，メジャーなサウンドボードには軒並み対応している。「Indy 500」の流れを汲むだけあって，ライバル車のエンジン音もステレオで鳴り響く。

正しくドライビングシミュレータであるから，ストイックにセッティングを煮詰めてタイムアタックするような遊び方もＯＫ。「インディ＝オーバル」と思われがちであるが，実際はスーパースピードウェイ(2.5マイルオーバル)やショートオーバル(１マイル)，ストリートコース(Ｆ-１みたいに曲がりくねったコース)など，いろいろと個性を持ったサーキットを転戦するシリーズである。この「INDY CAR RACING」にもいくつかのコースが入っている。もちろん作り込みは尋常でない。ラグナ・セカに行ってコークスクリューという名物コーナーも走ることができる。

ただひとつ難点を挙げるとすれば，セッティングをうっとうしい階層メニューにしてしまったこと。「Indy 500」では，練習走行中は走りながらセッティングを変えられるという大胆きわまるゲームデザインに感服したものだ。タイムアタックしながらギヤ比やウィング角を変え，ピットストップすることなくセッティングが煮詰められるのがとても気持ちよかったのだ。「WORLD CIRCUIT」の模倣か，それとも日本製ゲームの悪影響か。「INDY CAR RACING」が階層メニューで操作性を大幅に落としてしまったのはいただけない。

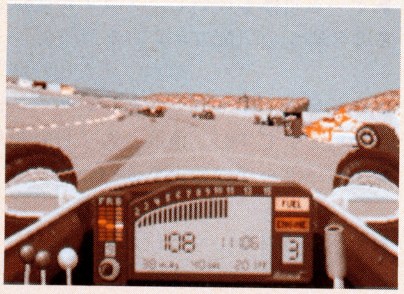
＊　＊　＊

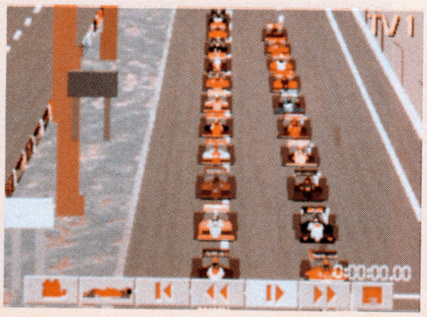
アーケードゲームのような専用システムでもなく，グラフィックワークステーションのようなコストを度外視した高級システムでもなく，そしてAMIGAのような気のきいたグラフィックアーキテクチャでもなく，ほとんどCPUパワーだけが武器といっていい汎用コンピュータの上でこれが実現されていることに時代の流れを感じる。

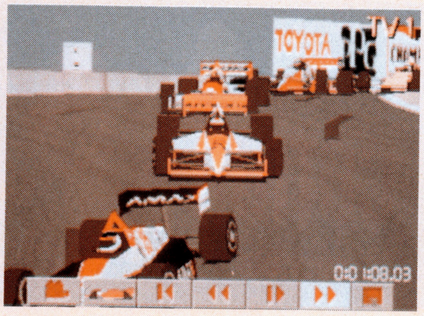
私は，3DOにこういうソフトが登場してくれることを願っていた。リアルタイムのテクスチャマッピングはこういう用途にはうってつけだ。が，現状を見ているとどうも心配。値段が値段なのだから，ちゃんとオトナでも長く遊べるきちんとしたソフトを出していかないと，勘違いマルチメディアマシンのひとつになりかねないのだ。

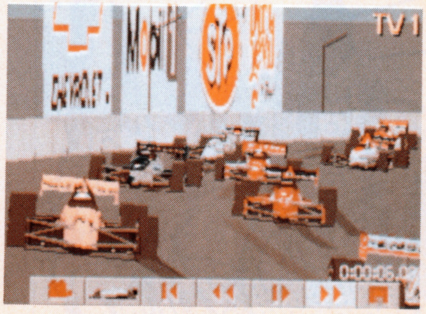
ちょうど悪いことに，私はそこそこ高速なDOS/Vマシンを即金で買えるくらいのお金を持っているので，そっちに転んでしまいそうである。「WORLD CIRCUIT」の驚異的に滑らかな動きにも魅かれてしまっているのだ。ああ。　(A.T.)

制作　PAPYRUS
発売　Virgin









AFTER REVIEW

アーケード版にあった問題点を解消し，X68000用により高い完成度で移植された「コットン」。ゲームの雰囲気，パソコンユーザーを対象にした丁寧な作りもずいぶんとユーザーの心をくすぐったようです。



コットン

▶こんなゲームがゲームセンターで埋もれていたなんて信じられない。
黒田　博明(24)富山県

▶キーボードは光るし，サンプリングは使いこなしているし，何回やっても飽きない。
宝福　公司(25)北海道

▶YOU DO！は私の手に負えません。
信太　徹(23)神奈川県

▶ひたすら敵が固い。こいつは「スターフォース」よりハードだ。　伴　武士(22)千葉県

▶よい，それだけ。でもEASYでもなかなかむずい。　主藤　二裕(25)福岡県

▶普通の人でも遊べる難易度，ボスの死神などが改良されているところは，よりよいゲームにしようとしているのがわかる。
中村　大輔(25)群馬県

▶パソコンのゲームソフトらしいオプション対応に好感がもてます。
佐竹　一生(27)福岡県

▶EASYなら初心者でもクリア可能（コンティニューすれば)。YOU DO！だと上級者でもクリア不可能(?)という絶妙な難易度設定がいい。　植木　正幸(24)神奈川県

▶最近，「コットン」にハマっています。弟とハイスコアを競っているのですが，私がノーコンティニュークリアを果たすと，弟はアイタタ地蔵を破壊して5,000点稼ぐという荒技を発見したり，の繰り返しです。結局は，いかにティータイムでスペシャルボーナスに成功するかなんですけど。現在のハイスコアは152万点で止まっています(理論的には160万点を超えるはずだけど)。
広瀬　良一(22)茨城県

▶ハードディスクに対応しているし，あっちこっちにスクロールするし，いっくぽーんだから。　益子　暁(19)東京都

▶エンディングの空中遊泳するコットンを見てさらに感激しました。
奥村　真明(20)千葉県



▲相沢栄樹(東京都)

▶う〜む，湯飲みが予約特典だと知ったのは発売日(涙)。　井村　英二(22)滋賀県

▶なんといっても地面に当たっても死なない！　私はこんなシューティングゲームを待っていた。こう書くと，普段どんな死に方をしているか一発でわかってしまう。どじな私。　池田　譲太(25)大阪府

▶キャラクターが最高。でこぼこ漫才みたいなやりとりが好き。
高橋　竜次(24)青森県

▶かなりいい出来。アーケード版は知らないけど，すごく面白い。
松本　勝正(19)富山県

▶ビッグでグレイトだから。
河合　竜次(19)岐阜県

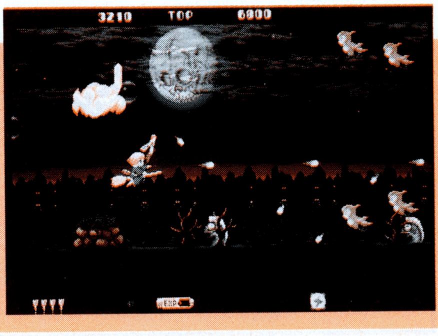
▶ゲーム自体もいいし，スコアが記録されるなど細かい気配りがいいと思う。
鈴木　正人(22)埼玉県

▶かわいいです。でもかわいいゲームはみんな難しいですね。蓮池　香子(19)福島県

▶ストーリーがゆかいで面白い。
寺本　公昭(25)熊本県

▶もう１カ月もハマり続けています。
美崎　善之(23)大阪府





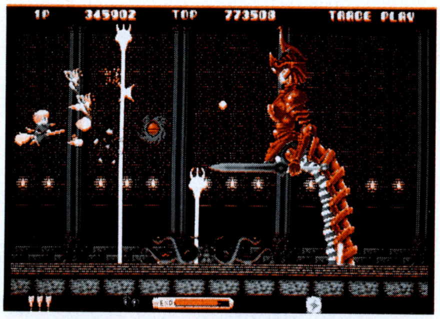
▶キャラクターとゲーム難易度とのアンバランスがいい。　小川　毅(20)埼玉県
▶アーケード版では，なかなか先に進めなかったけど，X68000版は遊びやすくなって結構先の面までいけるのがいい。まだ，ラストまではいってないけど。
中島　剛(30)福岡県
▶素晴らしい完成度。久びさにクリアするまでプレイしました。堀　幸司(24)福岡県
▶久びさに手応えのあるゲームだと思った。
田所　広行(20)茨城県
▶関係ないけど，コットンを買うとき店員さんはHゲームだと思っていたようだった。むう。さて，HARDモードはそれなりに難しいけど，EASYならばそれほどでもなく1,2回コンティニューすれば，エンディングまで見れてしまう。実際，「出たな!!　ツインビー」と同じようにソフトはよくできているけど，すぐにエンディングが見れちゃって空しい病にしばらくかかってしまいました。また，何回コンティニューしてもハイスコアはハイスコアだし，う～ん。
及川　剛(22)神奈川県
▶グラフィックの美しさといい，難易度といい，出来自体もすべてよし。
今井　彰彦(28)大阪府
▶いい意味でX68000用にアレンジされている（基板よりいいと思う)。
山之内　毅(23)福岡県
▶シューティングゲームの王道をいく単純さとバランスのよさ。努力すれば1コインクリアできる難易度は2重丸。
佐藤　貴是(22)神奈川県
▶アーケード版より面白い。エンディング(スタッフロール)があるのもいい。
松井　博明(23)群馬県
▶キャラクターがとってもいいです。でも難しい。　田中　和博(18)熊本県
▶何度もプレイしようという気にさせる適度な難易度がいい。久保田　学(24)愛知県
▶ゲームセンターでやり込む前に姿を消してしまい残念に思っていたので，X68000用に発売されて嬉しい！　面白いけど難しい。
大塚　正宏(20)千葉県
▶アーケード版より操作がやりやすくていいぞ～。　石田　伯仁(20)神奈川県
▶ゲーム構成がいい。音楽に合わせて光りまくるキーLEDは面白い。
松原　直人(34)福岡県
▶休んでぽん，がいい。
吉川　和男(22)東京都
▶全体的にバランスもいいし，なによりも雰囲気がよい。　青木　謙(19)神奈川県
▶やはり，いっくぽ～んでしょう。踊るキーLEDも驚いた。　金子　卓司(19)新潟県
▶オリジナル（アーケード版）より面白いから。　新井　雄一郎(19)宮城県
▶可愛さ余って難しさ100倍！
大平　浩貴(20)埼玉県
▶よくしゃべるし，グラフィックやアニメーションもきれいです。
小池　克博(23)山梨県
▶ただひたすらに撃ちまくる爽快感がたまらなくいいね。みょ～なノリのビジュアルも，おちゃめなPCMも雰囲気作りも僕の趣味に合っているようだし，遊んでいて非常に楽しい。EASYだとなんの気なしに遊べるので，暇があるとちょくちょく立ち上げてしまうんだよね。スコア稼ぎとかもかなり細かく随所にちりばめられているため，マニア受けもいいのだろう（個人的にはあまりこだわらないけど)。やっぱりシューティングゲームはこうでなくちゃ，と気に入っている作品だ。
（浜崎正哉）



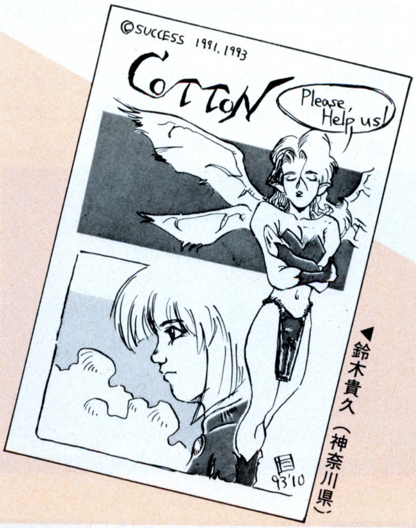
▲岩瀬貴代美（福岡県）



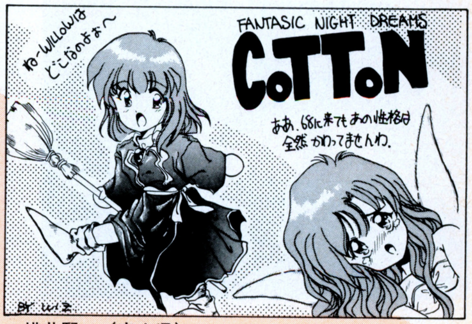
▲占部哲彦（広島県）







▲鈴木貴久（神奈川県）



▲横井賢一（富山県）

THE USER'S WORKS

●DarkElf

今回はゲームではなく，ゲームを作成するためのシステムを紹介する。アドベンチャーゲームやロールプレイングゲームを作成するための言語とシステムツール群だ。68ページの豪華マニュアルも付属している。

でーすれシリーズやPメーカーでお馴染みのT&H PROJECTS豊中支部からアドベンチャーゲーム記述言語が発表された。

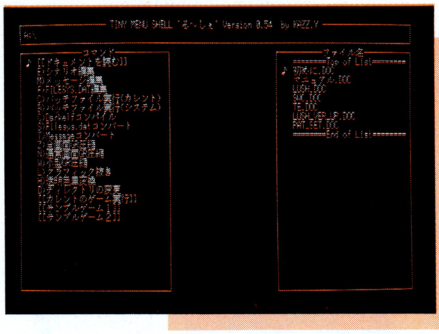
それまでこのサークルで使用していたロールプレイングゲーム用のイベント記述言語ELF(Event Language 1st.)に対して，アドベンチャーゲームの複雑なシナリオ管理に耐えられるようにすべくC言語ライクな言語仕様を取り入れ,「スパゲティプログラムにも対応できる」ようにしたのが今回のDarkElf (Deep Adventure's Regular Kernel of ELF) ということになる。

これさえあればなんの苦労もなくアドベンチャーゲームができる……というものではないが，マニュアルでは言語仕様以前にアドベンチャーゲームの作り方を手順を追って解説しているので，それを参考にすれば比較的簡単にアドベンチャーゲームが作成できる。

プログラミングの知識は，多少はあったほうがいいが，たいした問題ではない。ちゃんとシナリオを組んで，グラフィックを揃えて，音楽を用意して……という下地が揃ってさえいれば，シナリオどおりにそれらの関係を記述していくだけで完結したゲームの実行ファイルを作成してくれる。

DarkElfはコンパイラである。テキストでソースを書き，コンパイルして使用することになる。

ソースファイルはC言語のものと似ているが，ポインタや構造体といった複雑なデータ構造は現れないのでX-BASICしか使ったことのない人でもあまりとまどうことはないだろう。





特徴としては，

C言語ライクな制御構造

シーン管理ができる

コマンドメニュー管理ができる

ということだろうか。フラグなどの扱いは通常の言語の変数とさして違いはない。

シーン管理は当然の機能だろう。DarkElfでは1場面にあたる「カット」とそれをまとめた「シーン」の2つでシナリオの流れを管理している。

シーンごとにメニュー項目を記述しておくことによって，自動的にメニュー表示と入力処理，それにともなう各種処理の振り分けを行える。

ただし，メッセージやグラフィックは番号で管理されているので，事前に十分な基本設計を行っておくことが必要である。

基本言語仕様と出力関係は分離されており，表示その他の機能は外部関数によって実現される（ちなみに入力部分は言語仕様に取り込まれている)。

関数にはさまざまなものが用意されているのだが，超初級10関数が用意されているので最初はそれだけを把握しておけばいいだろう。

超初級といっても，簡単なアドベンチャーゲームやデジタルコミックの類はこれだけで間に合うくらいの機能は備えている。

各関数は超初級，初級，準中級，中級，準上級，上級，最上級といった区分分けされている。ちなみに最上級はメモリ確保などの関数だ。

サンプルゲームが2種類入っているので，それらのソースファイルを参考にするとよいだろう。

X68000/X68030に対応でメディアは5インチと3.5インチフロッピーディスク版が用意されている。

このソフトを入手希望の方は，宛名シール，2,000円分の無記名定額小為替(送料込み)，別紙にソフト名（必ず）とメディア種類を明記したものを同封して下記住所まで郵便で連絡してほしい。

〒560　大阪府豊中市本町8-6-28
T&H PROJECTS 豊中支部　前川　滋

プログラムの例

```
/* basic.h を使用する場合はまず最初にすべてで使うflagを宣言します */
flag    f1;

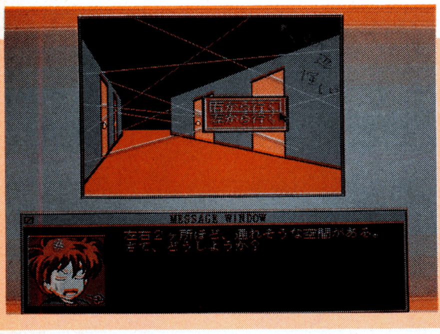
/* おまじない(初期設定をしてくれます) */
#include    basic.h

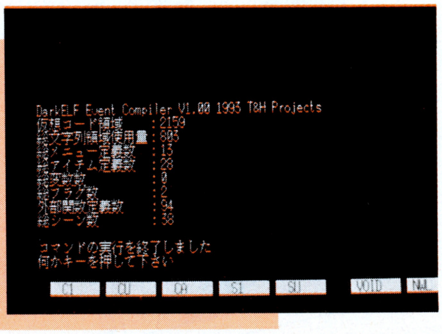
/* 最初のシーン名は必ず "start" です */
start   {
    GR_MAIN(FN_GR_G12);     /* メインCG */
    GR_SMALL(FN_GR_MIA1);   /* メッセージ横のCG */
    MES_PRINT(M_1_1);       /* メッセージ表示 */
    menu    {               /* 選択 メニュー */
    "まる":       scene   s2;   /* シーンs2へ */
    "ぺけ":       scene   s3;   /* シーンs3へ */
    }
}

s2    {
    f1 = 1;                 /* フラグf1を1にする */
    scene    s3;
}

s3      {
    GR_SMALL(FN_GR_MIA1);
    MES_PRINT(M_2_1);
    if (f1 == 1) {          /* もし、f1が1ならば */
        GR_SMALL(FN_GR_MIA2);
        MES_PRINT(M_3_1);
        GR_MAIN(FN_GR_G9);
    } else {                /* もし、f1が1でないならば */
        GR_MAIN(FN_GR_G19);
        GR_SMALL(FN_GR_MIA3);
        MES_PRINT(M_4_1);
        GR_MAIN(FN_GR_G16);
        WAIT60(60);             /* 1秒待ちます */
        GR_MAIN(FN_GR_G4);
        GR_SMALL(FN_GR_SYU1);
        MES_PRINT(M_5_1);
    }
    scene    ending;
}

ending  {
    GR_SMALL(FN_GR_MIA2);
    MES_PRINT(M_6_1);
    GR_MAIN(FN_GR_G10);
    MES_PRINT(M_7_1);
    END();                  /* これを忘れると暴走するよ */
}
```





［特集］

X-BASICとグラフィック

X-BASIC

GRAPHIC

今回は，X-BASICでグラフィック資源をどのようにして扱うか，X-BASICでのプログラミングの基礎をテーマとしてみた。
プログラミングをするうえで，グラフィック資源を使うだけではなく，コンピュータを使うためには，その資源に対する知識が要求される。かといって，カスタムICがどうのこうの……というようなレベルまでは必要なく，そのハードによってなにができるかを知っているだけでいい。
さらに，使おうとする言語がそれらのハードウェアのどこまでサポートしているかも把握しておこう。それらの要素を知ることによって，初めてどのようにして目的のものを作り上げていくかを考えられるのだ。
もちろん，それぞれの言語には特性があり，向き不向きは当然ある。特にX-BASICでは，実行速度の面でかなりのハンデを背負っている。しかし，X-BASICに合ったプログラミング，つまり，言語仕様からプログラムの仕様を考えるというアプローチもあっていいのではないだろうか。
限界だ，とすぐにあきらめずに，視点を変えてプログラミングに望んでみよう。きっと，実現可能な方法がどこかに隠れているはずだ。

CONTENTS

基本事項のおさらい

X-BASICに触れてみる

Hamazaki Masaya 浜崎 正哉

X-BASICへの思い入れタップリの浜崎氏が贈る簡単なプログラミングの話から，X-BASICをおさらいしてみましょう。欠点もいろいろあるX-BASICですが，まずは，ちょっとでもいいから触ってみませんか？

プログラミングって？

```
10 PRINT "HELLO"
20 GOTO 10
```

さあ，このプログラムはなにをするプログラムだろうか。Oh!Xの読者に対して思いっきりバカにした質問だが，もしも，わからない人がいたならさっそくX-BASICを立ち上げて実行し，マニュアルをめくって上記のリストにある命令を確認してほしい。ごく基本のお遊びプログラムということで誰しも一度は実行した覚えがあるだろう。導入部にありがちなサンプルなのでこれ以上解説せずに，次のリストを紹介する。

```
10 INT I,ANS=0
20 FOR I=0 TO 10
30   ANS=ANS+I
40 NEXT
```

いわゆる，0～10までの数を足していくプログラムだ。これも先ほどのリストと同様に自分の目で確認してもらいたい。特に，FOR～NEXTループの制御命令がどういったものかを理解するのが重要だ。では，このリストで求めた結果を表示するようにしてみよう。

これも特に問題はない。最初のリストで出てきたPRINT命令を使えばすむことだ。つまり，

```
50 PRINT ANS
```

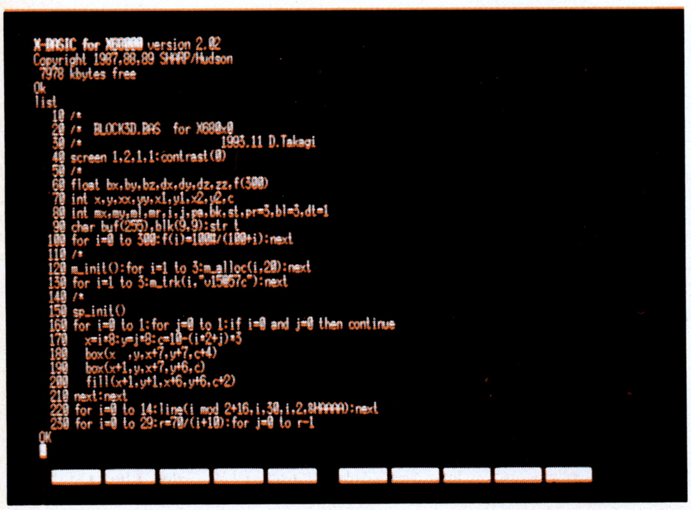
と1行追加すれば求めた結果が画面に現れる。

ではこれをさらに拡張していく。今度は，0～任意の数までの和を求めるようにする。そのためには，キーボードから数値を入力する必要があるのは明白だ。再び，マニュアルをめくって適当な命令がないか探してみよう。

すると，任意の数値を入力することができるINPUT命令を見つけられるはずだ。で，新しく追加して完成したリストが，

```
10 INT I,ANS=0,E
20 INPUT E
30 FOR I=0 TO E
40   ANS=ANS+I
50 NEXT
60 PRINT ANS
```

以上のようになる。FOR命令の最終値が入力した数値になっているところに注目してもらいたい。



X-BASIC起動画面

さらに，任意の数～任意の数までの和を求めるプログラムにしてみる。これはINPUT文を1行加えて，その入力した値を30行にあるFOR命令の初期値にすればいい。追加，変更は以下のとおり。

```
10 INT I,ANS=0,E,S
15 INPUT S
30 FOR I=S TO E
```

これも問題を解決するまでに時間はかからないだろう。

ところが，ここで開始値と終了値を逆転して入力した場合はどのようになるだろうか。すでにお気づきの方もいるだろうが，このままでは入力を間違うと正しい結果は得られない。新たに，

開始値＜終了値

となるような条件を満たしているかの判定を行う必要がある。判定を行うための命令は，もちろんIF文だ。これもマニュアルをめくって命令の仕様を確認してみよう。

で，条件が満たされない場合は，入力された開始値と終了値を入れ替える処理も加えておかなくてはならない。それらを考慮すると，

```
11 INT W
25 IF S>E THEN {
26   W=S:S=E:E=W
27 }
```

以上のリストを加えれば完成となる。なるべくなら，入力された数値が間違っていたら，自動的に修正されたことをメッセージで表示するとさらに親切かもしれない。これは，必要に応じて各自つけ加えるといいだろう。

ということで，急ぎ足でプログラミングの基礎の基礎＋マニュアルの基礎編レベルの話をさらりと解説した。特に問題なく読みこなせたらしめたもの。さらなる応用に向けてリストをいじくって遊んでみてほしい。

もっと楽しく

さて，いくら基礎的なこととはいえ，さすがに，0～10の数の和を求めるだけでは面白くない。実際，書いている僕としてもつまらないので，ちょっとゲーム寄りの話へもっていこう。

皆さんご承知のとおり，X68000には多彩な画面モードに65536色の表現力をもつグ

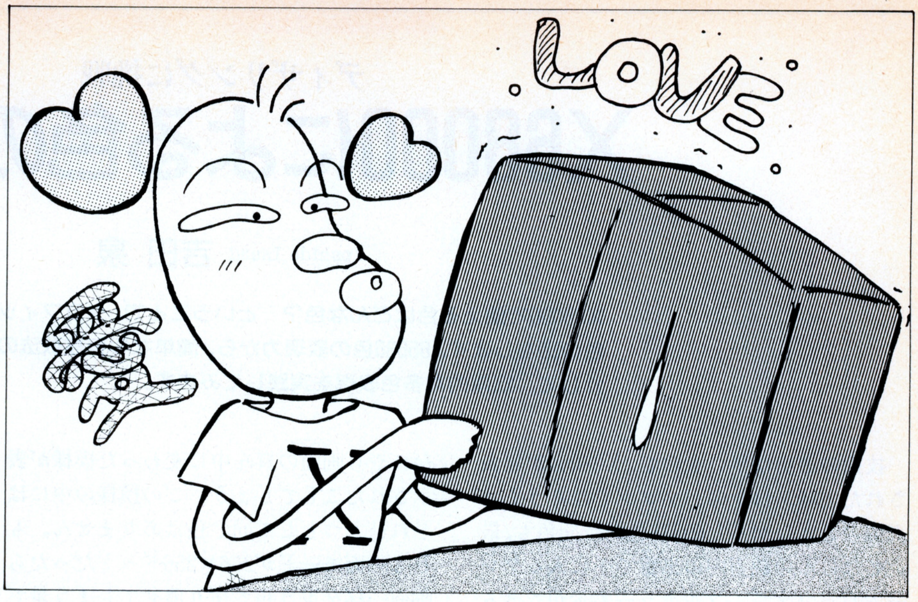
ラフィック，スプライト，ビットマップのテキスト画面まで用意されている。もちろん，X-BASICでそれらのおいしい機能を簡単に利用する命令がある（これは，マニュアルをぱらぱらめくっているだけで容易に確認できるだろう）。

実際に，ゲーム，そしてツールを作ろうとするとこれらの機能のありがたみが非常によくわかる。グラフを描くなら，やはり解像度が欲しいので768×512ドットモードを使う。ゲームなどでは多画面モードが重宝する。グラフィックを描きたいなら，豊富な色数＋適度な解像度の512×512ドットモードがいい。

特にゲームでは，スプライトなる非常に便利なキャラクター表示をするためのハードウェアまでサポートされているから，嬉しいったらありゃしない。現に，(で)のショートプロぱーてぃでは，X-BASICのみを使ったゲームが多数発表されているし，使い勝手は悪くないはず。

やる気があれば，実行速度を補うためのテクニックのひとつであるパレットアニメーションを駆使した，3Dタイプのゲームまで作ることができるしね。

サポートしていない機能も，ユーザーにそのオリジナル関数の作成方法まで公開されており，やろうと思えばなんだってできる気がするという，ちょっとだけ考えると非常に頼もしい仕様だ（深く考えて突っ込んではいけない）。



そして，壁にぶつかる

さんざん持ち上げるだけ持ち上げといて落とすのもなんだが，やはりX-BASICにも欠点はいろいろとある。

僕自身，あまり言語仕様などに精通しているわけではないので断定するのは危険かもしれないが，確かにプログラミングの入門にX-BASICは適している（理由はひと昔前にいわれていたのと同じ）。

しかし，慣れやすいぶん，というわけでもないだろうが，すぐに限界が見えてしまうのだ。理由は簡単。ある程度プログラミングに慣れてくると，すぐに作成しているプログラムが肥大化する。

しかも，欲が出て「あれもやりたい，これもやりたい」となるだろう。するとあっという間にX-BASICだけでできる範囲を超えてしまうのだ。

そこで，少ない家計でやり繰りする主婦の気持ちになってしつこく使い続けると，また，さらに先が見えてくるのだが，たいていの人はそこまでたどり着かない。いやはや残念。

やっぱり愛……だね

以前にも何回か書いているように，僕は基本的にアセンブラ人間だ。それでもなにかちょっとしたものを作ろうとすると，必ずX-BASICを立ち上げる。ついつい手がX-BASICを欲しがってしまう。ここまでくると，便利だからとかX-BASICでしかできないから，という次元を超えてしまっていると思う。

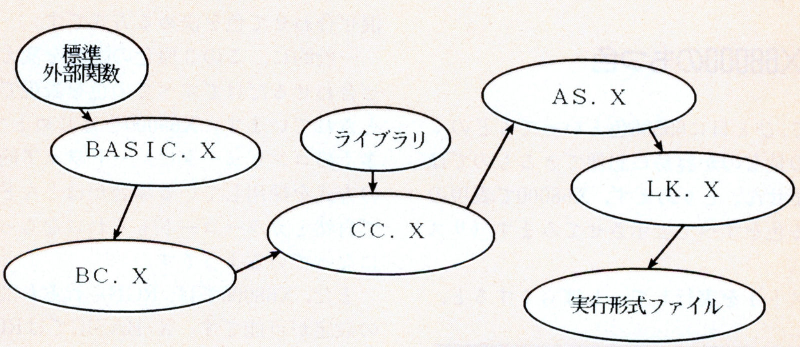
できの悪い子ほどかわいいとまではいわないが，ほとんど，幼少期に肌身離さず持ち歩いていたクマのぬいぐるみみたいな感覚だろう。なにしろBASICに対する愛着は，人並外れたものがあるはずだから（でなかったら説明がつかない）。

現在X-BASICを嫌っている人も，もう少し，X-BASICの立場になって使ってみると仲よくなれるかもしれない。

でもね，シャープさん。ええかげんバージョンアップしてくれないかなあ。いまさらBASICなんて……などと考えられるのはちょっと悲しいぞ。

X-BASIC & GRAPHIC

図　X-BASICのシステム構成



X68000の標準的な機能をひととおりサポートしているX-BASIC。言語仕様自体はC言語にコンバートできるという利点をもつ優秀な面もあるが，あくまで標準的な機能というところがくせもので，使えても損はないおいしい機能をサポートする命令がない，というのはなんともなさけない。特に半透明に代表されるグラフィック機能のサポートなど，変なところで抜けている（ファイル関係も弱いしね）。やっぱり，グラフィック，テキスト，スプライトのプライオリティの変更ぐらいはサポートしてほしかったね。これらは，外部関数で拡張すればすむことだが，もう少し，標準でかゆいところに手が届く命令が拡充されるだけでも，かなり使い勝手がよくなると思うのだが。いずれ行うかもしれないバージョンアップのときには，ライブラリの充実に期待したい（もちろんスピードアップもね）。

ディザリングに挑戦
X68000による色の表現

Yoshida Izumi 吉田 泉

X68000で使える色はどんな色？　ということで，グラフィックの基本である色の話をします。65536色の表現力から，簡単な誤差拡散法のプログラムを紹介して，疑似的な多色表現を実践してみます。

技術の進歩は早いもので，X68000が発表された当時，ほかのパソコンではそうそう真似のできなかった65536色同時発色も，最近ではほんの数万円の出費で手に入るフレームバッファで，フルカラー表示ができるようになりました。

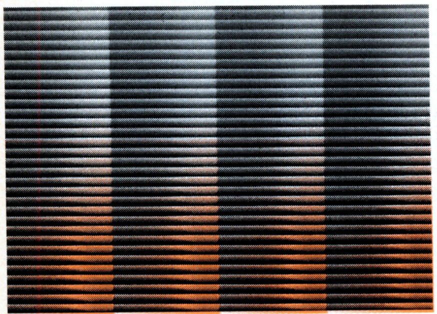
しかし，それらのボード類はほとんどWindowsや専用ソフト専用で仕様の公開もなく，とてもユーザーがプログラムを作って直接操作できるようなものではありません。その点，X68000のグラフィックで65536色同時発色が標準装備，というのは意義のあることでしょう。X68000があれば，誰でもほんのわずかなプログラムの知識さえあれば自由に操ることができるのですから（ちょっと大袈裟かな）。

このX68000にあるせっかくの機能を使わないテはありません。さあ，あなたもさまざまな色を自分でX68000に表示させてみましょう。そのときにはX-BASICがお手伝いをしてくれるでしょう。

X68000のもつ色

さて，ひと口に65536色といってもどのくらいの量なのか容易に想像できるものではありません。とりあえず，X68000で表現のできる色をすべて表示させてみます（リスト1）。

リスト1を実行して，しばらくすると，


これが65536色のすべてを表示したところ

なにやら画面の真ん中に変わった模様が表示されたことでしょう。この模様の中にはどれひとつとして同じ色はありません。もしも画面モードが256×256ドットだったら画面一杯を違う色で塗り潰せてしまう量です(画面モードについては，X-BASICユーザーズリファレンスのscreen命令の項に詳しい説明があるのでそちらを参照してください)。

今回は数ある画面モードの中でも最も豪華（？）な，512×512ドット，65536色モードを扱っていくことにします（screen命令でいうならば，screen 1,3,1,1 のモードですね）。

なおリスト1は単にすべての色をカラーコード順に並べて表示しているだけです。色を表す方法に光の3原色を用いるRGB式というのがありますが，この模様がその順番に並んでいるのがわかるでしょうか？　RGB式というのはその名のとおり，Red（赤）Green（緑）Blue（青）の3つの色を混ぜ合わせて色を決める方式です。

一般的に，この3原色の配分を変えて混ぜ合わせるだけですべての色を表現できるとされています。X68000をはじめとする，多くのコンピュータはハードウェア的にこの方式を採用している場合がほとんどなので自然とカラーコードもそれにならった順になっているようです。

また，X68000では，RGBそれぞれ32段階の設定が可能です。X-BASICではRGBからカラーコードに変換するrgb()という関数がありますが，緑の値を2048倍，赤の値を64倍，青の値を2倍したものを足し合わせることで同じことができることからもそれがわかります。

リスト1

```
10 screen 1,3,1,1:console ,,1:wipe()
20 int x,y,c
30 /*
40 c=0
50 for y=0 to 255
60   for x=0 to 255
70     pset(x+128,y+128,c)
80     c=c+1
90   next
100 next
```

もうひとつの色，HSV

さて，せっかくたくさんの色を表示させてもこれだけではちょっと芸がありませんね。また別の方法で色を表示させてみましょう（リスト2）。

なにやら円がいろいろな色で表示されています。これらの色はHSV方式（色相，飽和度，明度）で並んでいます。もっともS（飽和度）は変えていませんが。さて，現れた画面を見れば，V（明度）が円の中央から外に向かって大きくなるのはすぐにわかるでしょう。あとの色相は円の下側から反時計回りに増加する順に並んでいます。このことから，最初の色相は赤で，増加していくに従って紫，青，水色～に変化し，また赤に戻ってくるのがわかります。

なお，この方式はテレビ放送などに用いられています。まあ，過去の白黒放送と互換性をもたせるためにそうせざるをえなかったのですが。白黒放送では明度しか使っておらず，新しくカラー放送を行う際，その信号をまったく変えずに色をつけないと白黒テレビでカラー放送がまったく見えない，カラーで白黒放送が受信できない，というような問題が出てきてしまうからなんです。

このとき，なにを血迷ったかカラー放送にまったく違った方法，たとえばRGB方式などを利用していれば，テレビで色垂れなどが発生しないようにできたのですが，互換性の問題を考えるとどだい無理な話ですね。超豪華な白黒テレビを買った人たちが怒ってしまいますから。と，余談はこのくらいにして，つまり，テレビなどについている色合い調節ツマミが色相に相当したりします。

プログラムとしては特に変わったことはしていません。140行で円を表示するために三角関数を使っています。三角関数なんてわからないよ～，なんて声が聞こえそうですが，CGで使うぶんには角度と距離で点の位置を指定するようなもの，と思っていれば十分です。

角度は真下を0，反時計回りに増加していき1周したところ（360度回ったところ）で2π，X-BASICでは pi()×2 になります。これは，80行にありますね。そしてsin，cosがそれぞれX,Y座標になります。このままだと中心からの距離が1なので中心から離れたぶんだけ値を倍します。

140行ではjを0または63から127まで増加させて中心から外側に向かって扇型を描いています。このとき，明度を変化させているので中心のほうが暗い円が描かれるわけですね。130行でHSVの値をX68000のカラーコードに変換しています。説明の必要もないほどまったくそのままですね。このhsv関数の第2引数である31を減らすと，円全体の飽和度が上がり，だんだん白っぽくなっていきます。いろいろ値を変えて試してみましょう。

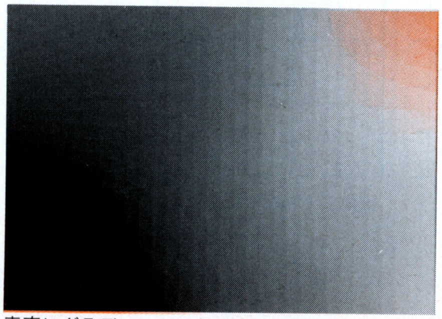
ところで，以上の掲載したプログラムたちはとっても遅いんです。ある程度は高速化（？）のようなことはしてあるのですが，X-BASICはやはり単純ループが苦手のようです（プログラムが悪い，とはいわないでね）。というわけで，BC（X-BASICからC言語に変換するツール。XCに付属）の使用をお勧めします。

CC /W ～.BAS

でコンパイルも問題なくできますしね。まあ，X-BASICで作業させてお風呂にでも入ってきてもいいのですが。

65536色は多いか，少ないか

さて，先ほどX68000では65536色も（？）使用できる，と書きましたが65536色というのは十分な色数なんでしょうか。



素直にグラデーションをかけてみた

まず，普通のCGなどでは65536色のすべて使ったものなんてお目にかかれませんよね。でも，各色，RGBごとに32階調ではちょっと不満があるでしょう。512×512ドットモードで画面一杯のグラデーションをかけたら，ひとつの色で16ドットも取ってしまいます。

これではあまりきれいなグラデーションとはいいがたいですね。そこで，まずはX68000に意地悪なグラデーションを表示させてみましょう（リスト3）。

あらら，プログラムを作った私でもがっかりするくらい予想以上に汚いグラデーション（？）ですね。このプログラムは40～70行で4隅の色をRGBごとに256階調で指定して，その間を滑らかに（？）つなごうというものです。色の指定がRGBごとに256階調，つまり，色の計算は256の3乗，16777216色でしています。いわゆる内部フルカラーというやつですね。この場合では右上を橙色（C1），左上を黄緑色（C2），右下を白（C3），左下を黒（C4）にしています。なんでこんな配置にしたかというと，以前私が絵画の先生に，絵の具を使う練習のためこんなものを描けといわれて苦労した経験があるからです。はは。

ではプログラムを見ていきましょう。まず170行までは単に2重のループで画面一杯に点を打っているだけです。180行からのget_color関数がミソ（というほどでもないか）ですね。

200～220行で各色をかけ合わせています。ここで，RGBそれぞれで4隅との距離に比例した色を計算しています。261161というのは511の2乗で，なんのことはない，511で2回割り算をする代わりです。そして230～250行で減色しています。ここでは単に，RGBそれぞれの最下位ビットを削っているだけですね。おかげでプログラムは簡単に作れますが，表示される色がRGBごとに境界線がわかるほど情けないものとなってしまいました。



もうひとつの色，HSV

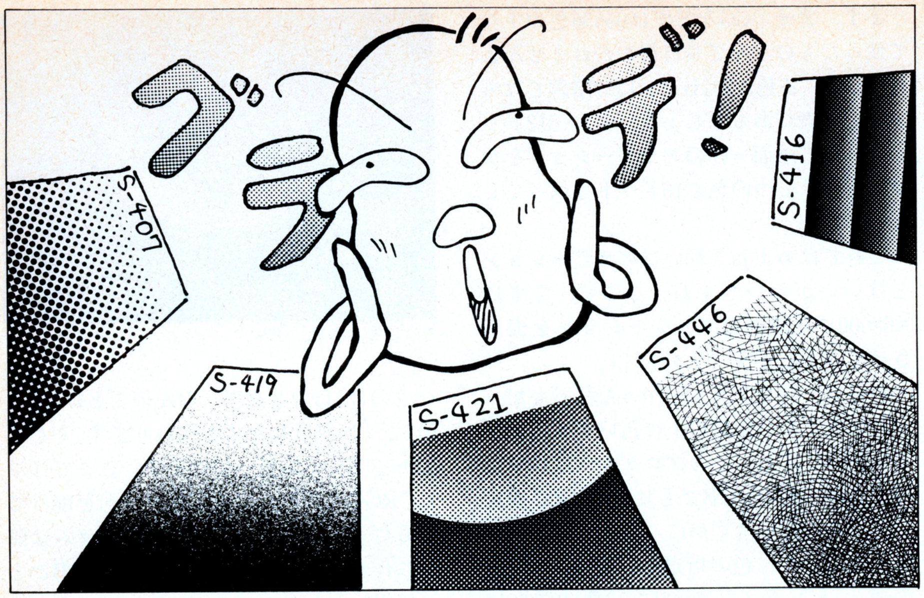
ちなみに内部フルカラーで各色256階調と設定したのは，一般的にそれ以上の色が，普通の人には区別できないらしいからです。まあ，世の中には普通でない人も多く見かけますので，もっと色が欲しい人はいるかもしれませんが。

リスト2

```
10 screen 1,3,1,1:console ,,0:wipe()
20 /*
30 int hu,va
40 int i,j,c,s
50 float r
60 /*
70 for i=0 to 2000
80   r=i*pi()*2/2000
90   hu=i*191/2000#
100  if i mod 2 =0 then s=0 else s=63
110  for j=s to 127
120    va=j/4
130    c=hsv( hu,31,va)
140    pset( sin(r)*j+256,cos(r)*j+256,c )
150  next
160 next
```

リスト3

```
10 screen 1,3,1,1:console ,,0:wipe()
20 /*
30 int x,y
40 int c1r=255,c1g=127,c1b=0
50 int c2r=0,c2g=255,c2b=127
60 int c3r=255,c3g=255,c3b=255
70 int c4r=0,c4g=0,c4b=0
80 int c
90 /*
100 for y=0 to 511
110   for x=0 to 511
120     c=get_color( x,y )
130     pset(x,y,c)
140   next
150 next
160 end
170 /*
180 func get_color( x;int,y;int )
190 int cr,cg,cb,c
200 cr=((c1r*x+c2r*(511-x))*(511-y)+(c3r*x+c4r*(511-x))*y)/261161
210 cg=((c1g*x+c2g*(511-x))*(511-y)+(c3g*x+c4g*(511-x))*y)/261161
220 cb=((c1b*x+c2b*(511-x))*(511-y)+(c3b*x+c4b*(511-x))*y)/261161
230 cr=cr/8
240 cg=cg/8
250 cb=cb/8
260 c=rgb(cr,cg,cb)
270 return(c)
280 endfunc
```



擬似的な多色化

昔のパソコンでは8色カラーが当たり前でした。そして，ユーザーは当然それで満足せずになんとかして少ない色数で，多くの色を再現できないかと試行錯誤を繰り返してきました。

X-BASICにはサポートされていませんが，X1にあった強力なタイリングペイントなどがその代表格ですね。この方法は違う色を交互，または特定のパターンで並べ，離れて見ると違う色に見えるといったものです。この技術の進歩（？）のおかげで，現在，PC-9801などで，とても16色とは思えないようなグラフィックが多数あります。

しかし，この方法も，あらかじめタイリングを行うことを前提にした絵でなければあまりきれいになりません。

そこで，ここでは単なる減色に向いている別の方法，ディザ法をやってみることにしましょう。

輝度ビット

本文でも触れているとおり，X68000の65536色モードは，RGB各32階調の表現が可能です。32階調というと，5桁の2進数で表現できますね。で，65536色モードのときは，1ドットごとに1ワード（16ビット）使われ，

ビット	15〜12	11〜6	5〜1	0
	G	R	B	

以上のようなビット構成になっています。

しかし，このままでは1ビット余ってしまいます。この残りのビットがどんな機能をもっているかというと，0ビット目は輝度ビットといわれ，階調を半段階上げるという役割をもっているのです。

この半段階というのが，結構扱いづらいもので，市販のアプリケーションは，輝度ビットをワークとして使うことが多いようです。実際に「Z's STAFF」や「MATIER」は，輝度ビットをマスク情報として利用しています。

ディザ法（dither method）

階調の多い画像を減色すると，リスト3で表示されるような，元の画像にはないはずの輪郭線が出てきます。これは偽の輪郭線といわれます。そこで雑音（というか，ノイズ）を加えて，その輪郭線を目立たなくする方法をディザ法（dither method）といいます。ディザ法にはパターンや，誤差拡散法などがあるのですが，ここでは最もアルゴリズムの簡単な誤差拡散法を扱っていくことにします。

まず，誤差拡散法のアルゴリズムについて簡単に説明しましょう。

画面などに点を打つ場合に，ハードウェアなどの制限により出力すべきデータと違う値を出力せざるをえません。そのときに，出力すべきデータと実際の出力したデータの間に発生する差を誤差といいます。

そして，その誤差をある適当な規則により周りの点に割り振っていくのです（図1）。そのあとに出てくるデータも同じように処理していくことによって，結果として全体ではすべての誤差を吸収できることになります。

ディザ法の実行

先のプログラムにおいて，誤差拡散法により処理を行ったものがリスト4のプログラムです。

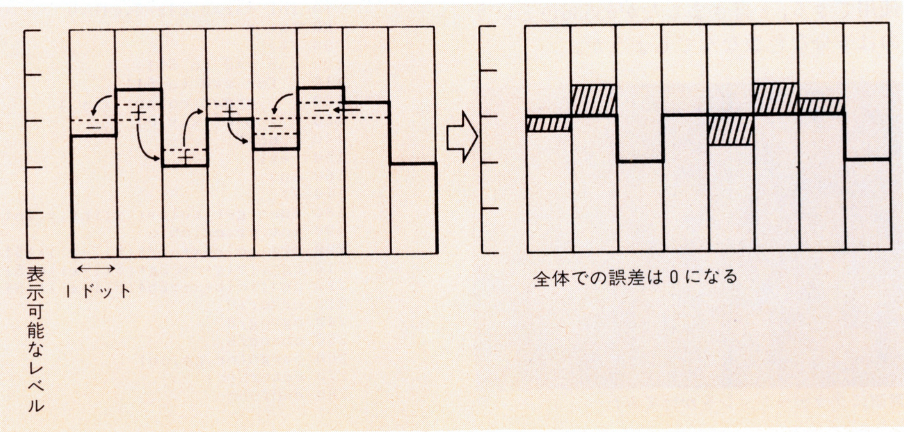
実行してみると確かにリスト3よりも多少はきれいなグラデーションになっているのがわかると思います。

まず，100〜110行ですが，これは次のラインへ拡散すべき誤差と，前の行から拡散されてきた行を入れる配列を宣言しています。120行では同様に次の（X方向）ドットへ拡散する誤差と前のドットから拡散されてきた誤差を入れる変数を定義しています。最初のラインは前のラインからの誤差は当然ないので，150行からは，配列をクリアしてます。

実際の誤差拡散の処理は340行から始まっています。まず始めにそれぞれの色を算出する部分は前と変わりませんが，その色に，前のラインと前のドットの誤差を加えています。そのあと，これから打つ点の誤差を取っています。ここでは削られるべき最下位ビットを余りとして求めています（380行から）。

そのあと，450行から各色ごとに誤差を周りのドットに拡散するように各変数に割り振っていきます。割り振りかたは，130行で宣言している配列erの内容によって振り分けられます。実数演算をして各ドットに数％ずつ割り振るのが本当なのですが，そん

図1





誤差拡散法によるディザリング

なことをしたら誤差を格納する変数が膨大になりますし，だいいち，速度が遅すぎます。そこでここではif文により，誤差の量に合わせて右，下，左下の優先順位で誤差を割り振っていきます（図２）。

ディザ法の結果

以上のように，ディザ法を実行することで，多少は色数を疑似的に増加させることができるのですが，プログラムの結果から見ても今回のディザ法は，まだまだ十分であるとはいえませんね。

ほかにも，パターンによる拡散を使ったディザ法とか（4×4ドットのパターンが一般的）を用いるとかすれば確実に偽の境界線を隠しやすくなります。もしくは，それよりも今回の方法で，きちんと実数演算し，きちんと誤差を拡散してもかなりの改善が望めるでしょう。

でも，今回の場合はどちらかといえば用意したグラデーションが広すぎたから，と逃げておきましょうか。普通の場合はこれで十分と思いますけれども。

見せかけの色

以上，X68000がもつ色はどんな色。てな具合で解説をしてみました。確かに65536色を数としてみるとそれほど不足はありませんが，やはり，RGB16ビット，32階調では不満が残ります。現在のグラフィック環境で標準となっているフルカラー画像を見てしまうとうらやましいかぎりですね。

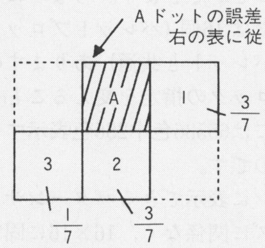
PC-9801などの世界でも長らく君臨していた16色タイルパターン文化から脱出しようとしていますし，コンシューマ機の世界ですらフルカラーが標準となってきているこのご時世。

せっかく，AV機能がウリとして登場したX68000なんですからがんばって進化してもらいたいものです。贅沢をいえば，標準で使えるとさらにいいんですけどね。

リスト4

```
 10 screen 1,3,1,1:console ,,0:wipe()
 20 /*
 30 int x,y,i
 40 int c1r=255,c1g=127,c1b=0
 50 int c2r=0,c2g=255,c2b=127
 60 int c3r=255,c3g=255,c3b=255
 70 int c4r=0,c4g=0,c4b=0
 80 int cr,cg,cb,c
 90 int cer,ceg,ceb
100 dim int n_lr(511),n_lg(511),n_lb(511)
110 dim int o_lr(511),o_lg(511),o_lb(511)
120 int n_r,n_g,n_b
130 dim int er(2,7)={0,1,1,2,2,2,3,3,0,0,1,1,2,2,2,3,0,0,0,0,0,1,1,1
}
140 /*
150 for i=0 to 511
160   n_lr(i)=0
170   n_lg(i)=0
180   n_lb(i)=0
190 next
200 for y=0 to 511
210   for i=0 to 511
220     o_lr(i)=n_lr(i)
230     o_lg(i)=n_lg(i)
240     o_lb(i)=n_lb(i)
250   next
260   for x=0 to 511
270     c=get_color( x,y )
280     pset(x,y,c)
290   next
300 next
310 end
320 /*
330 func get_color( x;int,y;int )
340 cr=((c1r*x+c2r*(511-x))*(511-y)+(c3r*x+c4r*(511-x))*y)/261161+o_
lr[x]+n_r
350 cg=((c1g*x+c2g*(511-x))*(511-y)+(c3g*x+c4g*(511-x))*y)/261161+o_
lg[x]+n_g
360 cb=((c1b*x+c2b*(511-x))*(511-y)+(c3b*x+c4b*(511-x))*y)/261161+o_
lb[x]+n_b
370 /*
380 cer=cr mod 8
390 ceg=cg mod 8
400 ceb=cb mod 8
410 cr=cr/8
420 cg=cg/8
430 cb=cb/8
440 /*
450 if cer<>0 then{
460   n_r=er(0,cer)
470   if x>0 then n_lr(x-1)=er(1,cer)
480               n_lr(x  )=er(2,cer)
490 }
500 if ceg<>0 then{
510   n_r=er(0,ceg)
520   if x>0 then n_lg(x-1)=er(1,ceg)
530               n_lg(x  )=er(2,ceg)
540 }
550 if ceb<>0 then{
560   n_r=er(0,ceb)
570   if x>0 then n_lb(x-1)=er(1,ceb)
580               n_lb(x  )=er(2,ceb)
590 }
600 /*
610 if cr>31 then cr=31
620 if cg>31 then cg=31
630 if cb>31 then cb=31
640 c=rgb(cr,cg,cb)
650 return(c)
660 endfunc
```

図2



Aの誤差	拡散する割合		
	1	2	3
0	0	0	0
1	1	0	0
2	1	1	0
3	2	1	0
4	2	2	0
5	2	2	1
6	3	2	1
7	3	3	1

今回のディザ法の拡散方法

スプライト基本知識のまとめ

X-BASICで学ぶスプライト・BG

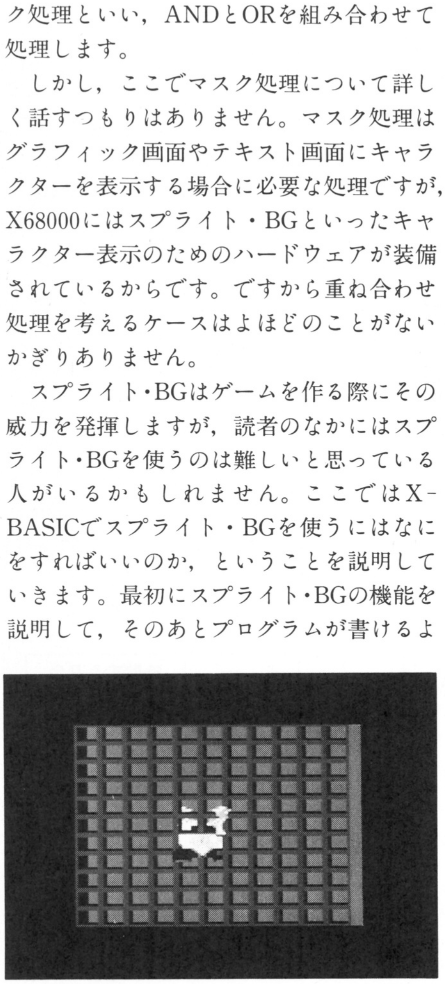
Asakura Yuji 朝倉 祐二

一見複雑そうに見えるスプライトも，実は非常に簡単に扱えるのです。マニュアルを読んだだけで挫折してしまった人は，この基本知識を読んでもう一度チャレンジしてみてください。

たとえばシューティングゲームを作るとしましょう。試行錯誤してゲームシステムを決め，キャラクターデザインを終え，プログラム作成の段階になります。ゲームではたくさんのキャラクターが表示されます。普通なら，キャラクターパターンは扱いやすさの点から16×16ドットなどブロック型に作るでしょう。パターンをブロック型で定義すると，あるキャラクターに別のキャラクターを重ねて表示するとき，重ね合わせ処理をする必要があります。これをマスク処理といい，ANDとORを組み合わせて処理します。

しかし，ここでマスク処理について詳しく話すつもりはありません。マスク処理はグラフィック画面やテキスト画面にキャラクターを表示する場合に必要な処理ですが，X68000にはスプライト・BGといったキャラクター表示のためのハードウェアが装備されているからです。ですから重ね合わせ処理を考えるケースはよほどのことがないかぎりありません。

スプライト・BGはゲームを作る際にその威力を発揮しますが，読者のなかにはスプライト・BGを使うのは難しいと思っている人がいるかもしれません。ここではX-BASICでスプライト・BGを使うにはなにをすればいいのか，ということを説明していきます。最初にスプライト・BGの機能を説明して，そのあとプログラムが書けるようにスプライト・BGを表示するサンプルプログラムを紹介します。最後まで読んだあなたは，スプライト・BGを難しいとはもう思わないでしょう。



サンプル実行結果

スプライト・BGってなんだ?

スプライト・BGの特徴を考えてみると，

・マスク処理を考える必要がない
・画面の書き換えが高速である
・水平，垂直反転表示機能がある
・BG，スプライト，テキスト，グラフィック画面間で表示優先順位を変更できる

などが挙げられると思います。これらの特徴は，大量かつ高速にキャラクターを表示することが要求されるシューティングゲームなどの作成に，スプライト・BGが非常に適したハードウェアであることを証明しています。

スプライトは表示サイズが768×512以外のとき使うことができます。X68000はスプライトを128プレーンもっていて，1ドット単位で表示位置を指定することができます。キャラクターの表示はプレーンにキャラクターパターンを指定し，そのプレーンをスプライト画面に表示することにより行います。もちろん複数のプレーンに同じパターンを定義することもできます。

さらに1つのパターンには1つのパレットブロックを指定して，ドット単位に16色を表示することができます。ちなみにパレットブロックは16個（パレットブロック0はテキストパレットと共通）ありますので，パレットブロックの指定を変えることにより，画面上には65536色中256色表示することができるのです。

1プレーンに表示できるパターンサイズは表示サイズに関係なく，16×16に固定されていて，プレーン番号が小さいほど表示優先順位が高くなっています。つまり同じ座標に複数のパターンが重なったとしても，プレーン番号の小さいものが上になるように表示されるということです。

またハードウェアの制約上，水平方向に33個以上のスプライトを表示することはできません。シューティングゲームなんかでキャラクターがたくさん表示されているときに，敵弾がちらついてうっきーな気分になってしまうことがありますよね。あれは水平ライン上に33個以上のスプライトを表示させたときに，一部のスプライトが表示されなくなるために起こる現象です。

そして，スプライトが1プレーンで16×16のパターンを表示するのに対して，BGは実画面いっぱいにパターンを表示することができます。ただし1つのパターンは表示画面サイズが512×512のときは16×16，256×256のときは8×8となりますので，どちらの表示画面サイズでも表示できるパターンは縦横32×32個分（実画面では64×64個分）となっています。

スプライト・BGに表示するキャラクターパターンは，PCGエリアと呼ばれるメモリ領域に記憶されます。PCGエリアはBG画面用エリアと合わせて32Kバイトあり，BGを使用しないときは32Kバイトのすべてを PCGエリアとして使用して，256個のパターンを記憶することができます。BGを1面使用するときは後半16Kバイトのうち半分の8Kバイトが，またBGを2面使用するときは後半の16Kバイトすべてが BG画面表示用に使われますので，最大登録パターン数はBGを1面使用すると192個，2面使用すると128個と減ってしまいます。

さて，スプライトとBGの用途ですが，何度もいうようにスプライトは1プレーンでは16×16ドットと表示できるサイズが小さいので，たとえば4プレーンをまとめて管理して32×32のキャラクターとして表示することが多いようです。

ただしハードウェアに複数のプレーンをまとめて大きなパターンとして管理する機

能はありませんので，あくまでプログラマが，ソフトウェアで管理を行ってやる必要があります。

一方，BGは実画面サイズいっぱいにPCGパターンを表示して，BG画面全体をスクロールさせることができますので，ゲームの背景によく使われます。また，画面の半分以上もあるような巨大キャラを表示させるときなど，スプライトのもつプレーン数で足りないときには,固定部分をBGに表示しておいて，可動部分をスプライトで表示させるといった合わせ技もあります。

キャラクターパターンを作ろう

スプライト･BGの表示はキャラクターパターンを作ることから始まります。キャラクターパターンを作るにも人それぞれいろいろなやり方があるでしょうが，ここでいくつかの方法を紹介しましょう。

A．SM.Xを使う

いわずとしれた横内威至氏制作のパターンエディタ。フルアセンブラで組まれたプログラムは高速。それだけでなく回転，拡大機能など機能も充実しています。しかもOh!Xの過去の付録ディスクに収録されたため当時の読者はわずか800円足らずで手に入れることができました。ちなみにSM.Xが収録された1992年6月号は現在入手不可能となっていますから，手に入れるには友人をあたるか，通信でも始めてX68000ユーザーの知り合いを作りましょう。

B．DEFSPTOOL.BASを使う

X68000のシステムディスクに付属のスプライトパターンエディタ。インタプリタで使うには処理速度に難点がありますから，コンパイルして使うことをお勧めします。機能的にはいまいちですが，SAVE機能はX-BASICでそのまま利用できる形でファイルに出力してくれるので，たまに使わせてもらっています。

C．市販のパターンエディタ，フリーウェアを利用する

市販のソフトウェアはお金がかかりますのであまり勧めませんが，フリーウェアにいいものがあればそれほどお金もかからないのでいいでしょう。また最近はX68000用にもフリーウェアをディスクに収録した単行本が多数発行されているようなので，それらの中から探してみるのもいいかもしれません。

D．方眼紙に色鉛筆で描く

パターンエディタと呼べるものを1つももっていなければ，方眼紙に16×16の四角形を描いてパターンを制作するよりしかたありません。実際に画面で表示させてみるとイメージが違うことがあり，また労力もかかりますが，退屈な授業中の暇潰しとしてはいいかもしれません。

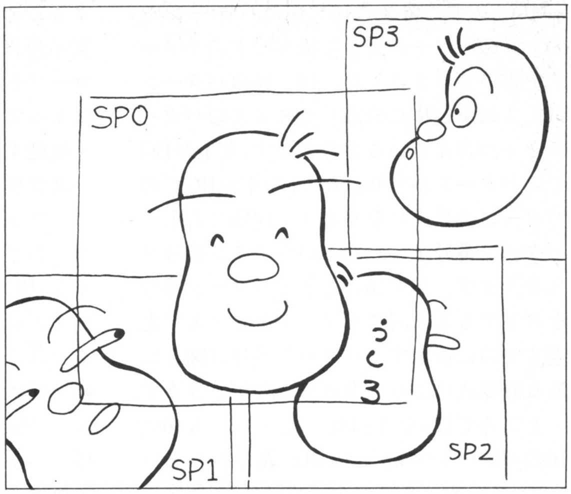


図1

スプライト画面の初期化・BGの設定
↓
PCGエリアにパターンを定義
↓
パレットブロックを定義
↓
スプライト・BG画面を表示
↓
スプライトの表示

スプライト･BGを表示しよう

スプライトを表示するには，大きく分けて図1に示した5つの処理を必要とします。ここでは図中の5つの処理とそれに対応したX-BASICでのサンプルプログラムリストを紹介しますので参考にしてください。X-BASICの各コマンドのパラメータの意味までいちいち説明しませんので，わからなければマニュアルで調べてくださいね。

・処理1
スプライト画面の初期化・BGの設定

スプライトを表示するにはスプライト画面の初期化を行わなくてはなりません。X-BASICではsp_initコマンド一発ですみます。ただしスプライトは表示画面が768×512のときは使えませんから，その前にscreenコマンドを実行して表示画面を512×512か256×256にしておかなければいけません（表1参照）。

またBGを使うにはbg_setコマンドを使います。bg_setの説明をマニュアルで見ると“BGにテキストページを割り当てます”といったようなことが書かれています。さてテキストページってなんでしょう？　前のほうでBG画面用エリアについて話しました。BGを使わないときはBG画面用エリアをPCGエリアに割り当てるので登録できるパターン数が増えるという話でしたね。

で,テキストページとはそのBG画面用エリアのことをいいます。テキストページ0がBG画面用エリアの前半8Kバイト,テキストページ1が後半8Kバイトに対応します。X-BASICでBGを使うのにメモリがどのように使われているかということまで知っておく必要はありませんが，覚えておいて損はありません。

・処理2
PCGエリアにパターンを定義

PCGエリアにパターンを定義するにはsp_defコマンドを使います。ドット単位に0～15のカラーコードの中から任意の1色を指定します。カラーコードはあとで説明するパレットブロック中のカラーコードに対応しますから，1つのパターンで65536色中16色を表示することができます。

定義できる最大パターン数はBGの使用状態によって変わってきます。表示画面サイズが256×256のときは，スプライトに定義するパターンサイズとBGに定義するパターンサイズが違いますので少し難しくな

図2

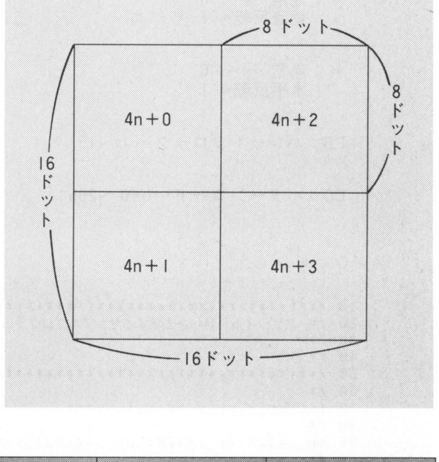


表1

表示サイズ	SP・BG使用可否	SPパターン	BGパターン	BG表示画面数
256×256	可	16×16	8×8	2
512×512	可	16×16	16×16	1
756×512	否	−	−	−

X-BASIC & GRAPHIC

っています。図2に16×16のパターンを8×8のパターンとして扱うときのイメージを書いておきました。16×16のパターンは，これを縦横に分割して8×8のパターンを4つ含んでいると考えます。左上がBGでのパターン番号0になり，左下がBGでのパターン番号1になります。同様に右上がパターン番号2に，右下がパターン番号3になります。16×16のときのパターン番号をnとすると，同じパターンを8×8で定義するのに指定するパターン番号は図2にある計算式によって求めることができます。

ところでBGを2面使ったとしても16×16のパターンを最大128個定義できるということは，8×8のパターンなら512個定義できるんじゃないの？　と考えた読者はなかなか鋭い。そのとおりなのですが，BG画面エリア内でパターン番号の指定が8ビット(表現できる範囲が0～255)になっているために，結局使えるパターン数は最大256個となっています。

・処理3
パレットブロックの定義

パレットブロックには，PCGパターンの設定の中で指定した0～15のカラーコードに対応するパレットコードを定義しておきます。X-BASICではsp_colorコマンドを使ってパレットブロックを定義します。このパレットコードは65536色の中から選ぶことができます。

パレットブロックは全部で16個ありますが，パレットブロック0はテキストパレットと共通のためにsp_colorコマンドで変更することはできません。スプライト・BGの表示色はPCGパターンに定義したカラーコードと，パレットブロックの組み合わせによって決まります。

・処理4
スプライト・BG画面を表示

スプライト・BG画面の表示はsp_dispコマンドで指定します。sp_disp(1)でスプライト・BG画面を表示します。スプライト画面を表示しておいて，スプライトの一部，または全部を表示，非表示するにはsp_on，sp_offコマンドを使います。BGの表示，非表示はbg_setコマンドの第3パラメータで指定します。

・処理5
スプライトの表示

スプライトを表示するコマンドにはsp_moveとsp_setがありますが，sp_moveコマンドはsp_setコマンドのパラメータの一部を標準的な値に固定したコマンドですので，ここではsp_setコマンドのみ説明します。sp_setコマンドの第4パラメータにはパターンデータを指定します。ここはX-BASICのマニュアルにもあるように，図3に示した複数の表示条件を設定する部分です。アセンブラを知らない人にとってbit(ビット)はわかりづらいでしょうから，図3に第4パラメータの値を求める計算式を紹介します。あとスプライト画面の座標は画面左上が(0，0)ではなく，(16,16)となっていますので注意してください。

図3

32768×V＋16384×H＋256×PB＋CD

V：通常………0
　　垂直反転…1

H：通常………0
　　水平反転…1

PB：パレットブロック…1～15

CD：パターンコード……0～255

最後に

さてひととおりスプライト・BGの使い方を説明してきました。なんとなくスプライト・BGが使えるような気持ちになってもらえたのなら嬉しいかぎりです。とりあえずなんでもいいからパターンを定義してスプライト・BGを表示してみてください。

サンプルプログラムは誌面で見るためだけに掲載しているのではありません。わけがわからなくてもいいですからリストを入力して実行してみましょう。うまく動いたらわからないコマンドをマニュアルで調べて，それでパラメータの値を変えて実行してみて結果の違いを理解するのです。

そうやってサンプルプログラムをどんどんいじって，スプライト・BGを自分のものにしていってください。

スプライトによるアニメーション

ゲームではアニメーションしているキャラクターが少なからずあります。PCGエリアに余裕があればいくつかのアニメーションパターンを定義しておいて順番に表示していけばいいでしょうが，実際には敵キャラクターやBG用のパターンなどもあるのでPCGエリアに余裕があるなんてことは滅多にないはずです。

PCGエリアが足りない場合のアニメーションの方法として，キャラクターを定義しているPCGエリアそのものを高速に書き換える，というものがあります。どれくらい高速にかというと，垂直帰線期間の間に書き換えを終了しないといけません。もしも間に合わないとパターン書き換えの途中で画面に表示されてしまって見苦しいことになります。

高速処理が絶対条件ですから，X-BASICのインタプリタでは無理です。コンパイラにかけるとしてもX-BASICには垂直帰線期間を検出するコマンドがないのでこれもダメです（Cに変換したソースに手を加えればできないことはないでしょうが）。残された方法としてPCGエリアを書き換えるプログラムを外部関数にしてしまう手があります。これなら速度的にも垂直帰線期間の検出にしても問題はありませんし，それにこういう外部関数があっても面白いんじゃないかな……ということで，このあと（この原稿を書き上げてから）そんな外部関数を作ってみようと思っています。

実は今月のサンプルプログラムで表示するキャラクターは，現在開発を進めているあるゲームの主人公キャラクターで，こいつにピョンピョンと跳ねるアニメーションをさせたいのです。

もしかするとこのゲーム専用の外部関数になってしまうかもしれないけど，さてどうなることやら。

ちなみに，今度の付録ディスクまでに完成して収録してもらう予定ですので，楽しみに待っててね。

リスト

```
10 /*************************************************
20 /* スプライト・BGを表示するサンプルプログラム
30 /*
40 /*                                  1994,Feb,Oh!X
50 /*************************************************
60 /*
70 int sp_x,sp_y,bg_x,i,w,cnt
80 /*
90 sp_x=128:sp_y=128:bg0_x=0:bg1_x=511
100 /*
110    init_sprite()        /* スプライト・BG画面の初期化
120 /*
130    set_pattern()        /* スプライト・BGパターンの設定
140 /*
150    set_palet()          /* パレットブロックの設定
160 /*
170    display_sprite()   /* スプライト・BG画面表示
180 /*
190 for i=10 to 20          /* BGの表示
200   for j=10 to 20
210       write_bg(0,i,j,0,0,2,20)
220       write_bg(1,i,j,0,0,3,20)
230   next
240 next
250 /*
260 /* 最後のパラメータはプライオリティの指定
270 /* 0～3まで値を変えて実行してみてください
280 /*
290    write_sprite(0,sp_x   ,sp_y   ,0,0,1,1,3)
300    write_sprite(1,sp_x+16,sp_y   ,0,0,1,2,3)
```

▶マルチエフェクタを一度使うとやみつきになりますぞ～。　眞田　二三男(21)東京都

```
310    write_sprite(2,sp_x    ,sp_y+16,0,0,1,3,3)
320    write_sprite(3,sp_x+16,sp_y+16,0,0,1,4,3)
330 /*
340 repeat
350    bg_scroll(0,bg0_x,0)    /* BG画面スクロール
360    bg_scroll(1,bg1_x,0)
370    bg0_x=bg0_x+1 and 511
380    bg1_x=bg1_x+2 and 511
390    if inkey$(0)=" " then {
400       while inkey$(0)<>"":endwhile
410       repeat:until inkey$(0)<>""
420   }
430 /* ESCが押されるまで繰り返し
440 until inkey$(0)=chr$(27)
450 end
460 /*
470 /*    *------------------------------*
480 /*   *- スプライト・BG画面の初期化 -*
490 /* *------------------------------*
500 func init_sprite()
510 /*
520      screen 0,3,1,1:sp_init()
530      bg_set(0,0,1):bg_set(1,1,1)
540 /*
550 endfunc
560 /*
570 /*    *------------------------------*
580 /*   *- PCGエリアにパターンを定義 -*
590 /* *------------------------------*
600 func set_pattern()
610 /*
620      dim char pat0(255)
630      dim char pat1(255),pat2(255)
640      dim char pat3(255),pat4(255)
650      dim char pat5(63)
660 /*
670   pat0={
680      +0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
690      +0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
700      +0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
710      +0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
720      +0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
730      +0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
740      +0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
750      +0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
760      +0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
770      +0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
780      +0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
790      +0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
800      +0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
810      +0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
820      +0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
830      +0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 }
840   pat1={
850      +0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
860      +0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
870      +0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
880      +0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
890      +0, 0, 0,13, 9, 9,13, 0, 0, 0, 0, 0, 0, 0, 0, 0,
900      +0, 0,15,15,15,10, 9, 9,13, 0, 0, 0, 0, 0,13, 9,
910      +0, 0, 0, 9,15,15,10,10, 9,13,11, 0,11,13, 9,10,
920      +0, 0, 0,13,15,15,15,10,10, 9, 0,11, 0, 9,10,10,
930      +0, 0, 0,15,15,15, 1,10,10,10,11,11,11,10,10, 9,
940      +0, 0, 0, 0, 9,15, 9, 8, 8,11,10,11,10,11, 8, 8,
950      +0, 0, 0, 0,13, 9, 8, 9, 9,10, 8,10, 8,11,10, 9,
960      +0, 0, 0, 0, 0, 9, 9, 9, 9,10,10,10,10,10,10,10,
970      +0, 0, 0, 0, 0, 9, 9, 9, 1, 1, 8,10, 9, 1, 1, 8,
980      +0, 0, 0, 0, 0, 9, 9,14, 1, 1, 1,10,14, 1, 1, 1,
990      +0, 0, 0, 0,13, 9, 9, 1, 1, 1, 1,10, 1, 1, 1, 1,
1000     +0, 0, 0, 0, 9, 9, 9, 8, 1, 1, 8,10, 8, 1, 1, 8 }
1010  pat2={
1020     +0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1030     +0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1040     +0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1050     +0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1060    +13, 9, 9,13, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1070     +9, 9,15,15,15, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1080    +10,15,15, 9, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1090    +15,15,15,13, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1100     +1,15,15,15, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1110     +9,15, 9, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1120     +8, 9,13, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1130    +10, 9, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1140    +10, 9, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1150    +10, 9, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1160    +10, 9,13, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1170    +10,10, 9, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 }
1180  pat3={
1190     +0, 0, 0, 0, 9, 9, 9, 9, 9,10,10,10,10,10,10,10,
1200     +0, 0, 0, 0, 9, 9, 9, 9,10,10,10,10,10,10,14,14,
1210     +0, 0, 0, 0, 9, 9, 9,10,10,10,10,12,10,10,14,14,
1220     +0, 0, 0, 0, 9, 9, 9,10,10,10,10,12,10,10,10,10,
1230     +0, 0, 0, 0,13, 9, 9, 9,10,10,10,10,10,10,10,10,
1240     +0, 0, 0, 0, 0, 9, 9, 9, 9,10,10, 9,10,10,10,10,
1250     +0, 0, 0, 5, 5, 5, 5, 5,13, 9, 9, 9, 9,10,13, 5,
1260     +0, 0, 5, 5, 5, 5, 5, 5, 5,13, 9, 9, 9,13, 5, 5,
1270     +0, 0, 5, 5, 5, 5, 5, 5, 5, 5,13, 9,13, 5, 5, 5,
1280     +0, 0, 5, 5, 5, 5, 7, 7, 7, 7, 6, 0, 6, 5, 5, 5,
1290     +0, 0, 5, 5, 5, 7, 7, 7, 6, 6, 0, 0, 0, 6, 6, 5,
1300     +0, 0, 3, 6, 6, 6, 6, 3, 0, 0, 0, 0, 0, 0, 0, 3,
1310     +0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1320     +0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1330     +0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1340     +0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 }
1350  pat4={
1360    +10,10, 9, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1370    +10,10, 9, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1380    +10,10, 9, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1390    +10,10, 9, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1400    +10, 9,13, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1410    +10, 9, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1420     +5, 5, 7, 5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1430     +5, 7, 7, 7, 5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1440     +5, 7, 7, 7, 7, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1450     +5, 5, 7, 7, 7, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1460     +5, 5, 5, 5, 5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1470     +6, 6, 6, 6, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1480     +0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1490     +0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1500     +0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1510     +0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 }
1520  pat5={
1530     +14,+14,+14,+14,+14,+14,+14,+14,
1540     +14,+1 ,+0 ,+0 ,+0 ,+0 ,+0 ,+14,
1550     +14,+1 ,+0 ,+0 ,+0 ,+0 ,+0 ,+14,
1560     +14,+1 ,+0 ,+0 ,+0 ,+0 ,+0 ,+14,
1570     +14,+1 ,+0 ,+0 ,+0 ,+0 ,+0 ,+14,
1580     +14,+1 ,+0 ,+0 ,+0 ,+0 ,+0 ,+14,
1590     +14,+1 ,+1 ,+1 ,+1 ,+1 ,+1 ,+14,
1600     +14,+14,+14,+14,+14,+14,+14,+14 }
1610 /*
1620     sp_def(0,pat0)    /* 16×16dotパターン定義
1630     sp_def(1,pat1):sp_def(2,pat2)
1640     sp_def(3,pat3):sp_def(4,pat4)
1650     sp_def(20,pat5,0) /* 8×8dotパターン定義
1660 /*
1670 endfunc
1680 /*
1690 /*    *------------------------------*
1700 /*   *-     パレットブロックの設定     -*
1710 /* *------------------------------*
1720 func set_palet()
1730     int i
1740 /*
1750     dim col1(15)={0,1,24,17168,48938,832,25812,
1760                  +1088,45856,65394,64822,29916,
1770                  +36844,39450,65534,36538 }
1780     dim col2(15)={0,0,0,0,0,0,0,0,0,0,
1790                  +0,0,0,0,6575,0 }
1800     dim col3(15)={3575,3575,0,0,0,0,0,
1810                  +0,0,0,0,0,0,0,3575,0 }
1820     for i=0 to 15
1830 /*
1840         sp_color(i,col1(i),1)
1850         sp_color(i,col2(i),2)
1860         sp_color(i,col3(i),3)
1870     next
1880 /*
1890 endfunc
1900 /*
1910 /*    *------------------------------*
1920 /*   *- スプライト・BG画面を表示 -*
1930 /* *------------------------------*
1940 func display_sprite()
1950 /*
1960     sp_disp(1)
1970 /*
1980 endfunc
1990 /*
2000 /*    *------------------------------*
2010 /*   *-  スプライトパターンを表示  -*
2020 /* *------------------------------*
2030 func write_sprite(pl;char,x;int,y;int,v;char,h;char,pb;cha
r,cd;char,pr;char)
2040 /*
2050 /* write_sprite(pl,x,y,v,h,pb,cd,pr)
2060 /*
2070 /* pl..プレーン番号
2080 /* x...X座標
2090 /* y...Y座標
2100 /* v...垂直反転(0..しない, 1..する)
2110 /* h...水平反転(0..しない, 1..する)
2120 /* pb..パレットブロック
2130 /* cd..パターンコード
2140 /* pr..プライオリティ
2150 /*
2160     sp_set(pl,x,y,v*32768+h*16384+pb*256+cd,pr)
2170 /*
2180 endfunc
2190 /*
2200 /*    *------------------------------*
2210 /*   *-      BGパターンを表示      -*
2220 /* *------------------------------*
2230 func write_bg(bg;char,x;int,y;int,v;char,h;char,pb;char,cd
;char)
2240 /*
2250 /* write_bg(bg,x,y,v,h,pb,cd)
2260 /*
2270 /* bg..BG番号
2280 /* x...X座標
2290 /* y...Y座標
2300 /* v...垂直反転(0..しない, 1..する)
2310 /* h...水平反転(0..しない, 1..する)
2320 /* pb..パレットブロック
2330 /* cd..パターンコード
2340 /*
2350 /*
2360     bg_put(bg,x,y,v*32768+h*16384+pb*256+cd)
2370 /*
2380 endfunc
```

▶ 1月号の町田華龍さんへ，公務員でもクビになることもあるよ。
熊井　保人(23)奈良県

プログラミングに触れてみよう

マンデルブロ集合を描く

Shibata Atsushi 柴田　淳

ここでは，非常に単純な式から生み出される摩訶不思議な図形として有名なジュリア集合，マンデルブロ集合をX-BASICで描きます。その理論をどのようにしてプログラム化するのかを，じっくり読み取ってください。

プログラミングのできないパソコンユーザーの多くは，プログラミングに対して「なにかとてつもなく難しいもの」という印象を抱いていないだろうか。しかし実際はそんなことはない。だいいちプログラミングがそれほど難しいものなら，「日曜プログラマ」と呼ばれる「趣味でプログラミングを楽しむ」ような人種が生まれるはずがない。

僕は数ある「創造的な」趣味の中で，プログラミングほど手軽なものはないと思っている。編み物だとか陶芸だとか，世の中には趣味と称していろいろなものを作っている人がいるわけだが，それらはもし失敗したとき，金銭的にも時間的にも，大きな代償を払わなければならない。たとえばセーターを編んでいて，途中から編み棒を持つ手に力が入って編み目が詰まってしまい，丈がツンツルテンになったときのことを考えてみるといい。

ところがプログラミングの場合は，プログラム全般にわたってある関数をタイプミスしてしまっても，エディタの置換機能を使えば一発で間違いを修復できる。単純な作業はほとんどコンピュータまかせにできるので，ハードディスククラッシュに備えてこまめにバックアップを取るなどの基本さえ押さえておけば，面倒くさい雑事に煩わされることはほとんどない。

しかし，プログラミングを始めようという人にとって「なにから始めるか」というのは大きな問題である。世の中の主流はC言語だ，という話をよく聞くので，コンパイラ一式とCの入門書を買ってくる。だが，画面に“Hello World”と表示できるようになっても，たいていの人は構造体やポインタの概念を自分のものにできずに投げ出してしまう。

アセンブラでそれなりにプログラミングの経験を積んできたような人ならこれらの概念は直感的に飲み込めるものだが，プログラミングに関して真っ白な状態から始めるユーザーにとっては，ポインタという概念がなぜあるのかすら理解できないだろう。そんな人は，「ポインタにつながれた構造体のメンバを参照するときはポインタ演算子をつけます」とかいう入門書の言葉は，意味はわかってもどうしてそういうものがあるのかが理解できない。多くの入門書には「どうして」の部分の説明が不足しているのが原因の一端である。

だいぶ話がずれてしまった。要するに僕がいいたいのは，世の中の主流であるC言語に近く，しかも手軽であるX-BASICはプログラミングの取っ掛かりとしてはなかなかよいのではないか，ということなのだ。ちなみにここで手軽というのは，エラーがあっても暴走しないとか，途中でプログラムを止めて変数を覗いたりできる，という意味である。

ところで，プログラミングがほかの趣味に比べ手軽だとしても，実際画面に向かってプログラムを書くとなると一定の思考を余儀なくされる。で，人間というのは打算的な生き物だから，苦労をしたならそれなりの代価を得られないと納得しない。要するに，「さあ，プログラミングをしましょう」といわれて苦労をしてプログラムを作っても，出来上がったものがチャチなジャンケンゲームだったりすると納得しないのだ。

プログラミングを始めようと思っている人に必要なのは，ひとつは手軽な言語。そしてもうひとつは，手軽で大きな見返りの期待できる題材である。そのような題材を見つけるのはなかなか難しいが，ないことはない。

まさかと思うかもしれないが，フラクタル図形を描くというのが，初心者にもってこいの題材なのだ。具体的なターゲットとしては，以下の条件に当てはまる読者である。

1)　中学3年程度の数学を理解できる
2)　X-BASICの初歩的な文法を理解できる
3)　小一時間集中できる

2番目の条件の「初歩的」の定義に関してだが，大雑把にいってX-BASICのマニュアルの「基礎編」を理解している程度で十分である。

また，文法的にわからないところが出てくれば，マニュアルを開いて確認をすればいい。プロのプログラマだって，わからないことを調べるため参考書などをしょっちゅう開くものなのだ。プログラミングに関して，すべてを覚えていなければならない，というわけではないのである。

一応サンプルのプログラムを掲載しておくが，これはあくまでも見本でしかない。あなたのプログラムがこれと同じである必要はないし，また読者に打ち込んでもらうためにプログラムを掲載するわけでもない。とにかく，恐れずに自分で作ってみることが大切である。

虚の平面

実際にプログラムを組んでみる前に，少しだけ数学的な話を。といっても，先に提示した条件どおり，中学3年程度の数学的知識で理解できる話をする。

次の2次方程式を解いてみよう。

$$x^2+2x+4=0$$

単純に解の公式を使って解くと，

$$x=2\pm\sqrt{-3}$$

となる。文部省の方針で，中学生の数学では中の数字がマイナスになるような平方根はないと教えることになっているので，先生は「このような場合は，2次方程式の解はありません」と教えてくれるのだが，高校に入ると，これは真っ赤な嘘であることに気づく。

この2次方程式の解のような，ルートの中がマイナスになってしまう数を扱うために，2乗すると−1になるような数iがあ

る。誤解を恐れず書けば，iとは$\sqrt{-1}$のことだ。したがって，$\sqrt{3}$ i というのは$\sqrt{3}\times\sqrt{-1}$であり，結局$\sqrt{-3}$と同じである。だから先ほどの方程式の解は，

$$x = 2 \pm \sqrt{3}\,i$$

と書けるわけである。

数学では，このiがくっついている項を虚数項と呼ぶ。またiは文字式の文字と同じように扱える。ただ気をつけなければならないのは，iを2乗すると−1になる，ということだけである。

ところで，iがくっついている項を虚数項と呼ぶわけだが，では$2 \pm \sqrt{3}\,i$のように，虚数項と実数項をもっている数は，虚数だろうか，それとも実数だろうか。だいいち，この式のように実数に虚数を足したり引いたりしてしまっていいものだろうか。



まず，2番目の疑問に対する答えは，「別にかまわない」である。iは文字式の文字と同じに扱っていいのだから，問題の数は$2+\sqrt{3}a$のような式と同じと考えてさしつかえないのである。

ただし，文字の交ざった式を文字式というのに対して，虚数項の交ざった数を複素数という。だから，1番目の「虚数項と実数項をもっている数は虚数か実数か」という疑問に対する答えは，「実数でも虚数でもない」となる。なお複素数というのはただの呼称だから，そういうものがあるのだと覚えていただくだけでよい。

ところで，実数を目で見える形で表す方法として，数直線というものがある。実数と同様に，虚数も数直線上に表すことができるわけだ。

で，実数と虚数を両方もつ複素数を表す場合，どうすればいいか。鋭い読者はすぐ気づくと思うが，実数値を横軸に，虚数値を縦軸に取れば，2つの数値をいっぺんに表すことができる。この平面を複素平面と呼ぶ。ものものしい呼び名だが，定義はそれほど難しくない。

自己平方数列の不思議

複素平面というものがある，と理解していただければ，フラクタル図形を描くところまであと一歩だ。先を急ごう。

ここで，話はがらりと変わる。たとえば1日お金を預けると1%の利子がつく銀行があるとしよう。この銀行に10,000円の預金をしたとすると，次の日には100円の利子がつき，預金は10,100円になっている。その次の日は元金10,100円に利子がつくのだから，101円を足した10,201円に預金額が増えているはずだ。

では，この銀行に預けたお金が，100日後にいくらに増えているかを計算するためには，どうしたらいいだろうか。単純な方法だと，

当日の預金額＝前日の預金額×1.01

という計算を100回繰り返せば，答えを導くことができる。また逆にいえば，100回の計算をしなければ100日後の利子は求められない，ということになる。このように，ある値を求めるために，そのひとつ前の値に数値をかけるような数列は，等比数列と呼ばれている。

そのほか，一定の値を足して次の値を求める等差数列などがあるが，ここで取り上げるのは自分自身を2乗し，さらに一定値を足して次の項を求める数列である。しかも，実数と虚数の交ざった複素数でこの数列を作るのだ。

いきなり話の内容が難しくなったような気になるかもしれないが，そんなことはない。複素数といえども，文字式と同じように扱えばいい。試しに，

$$(3+2\,i)^2$$

という計算をしてみよう。中学で習う展開公式を使えば，

$$9+12i+4\,i^2$$

となる。ここで気をつけてほしいのは，実数×虚数は虚数になるということだ。また，iは2乗すると−1になるのだから，この式は結局，

$$9-4+12\,i=5+12\,i$$

という複素数になる。ポイントは，複素数を2乗しても複素数のままだ，ということである。

先ほどの銀行の預金の等比数列の例では，数列は1回計算をするにしたがって増えていくばかりであった。では，複素数を2乗して定数を足す，という数列の場合はどうだろうか。実際に数列を何項か計算して，数列の増減を見てみよう。

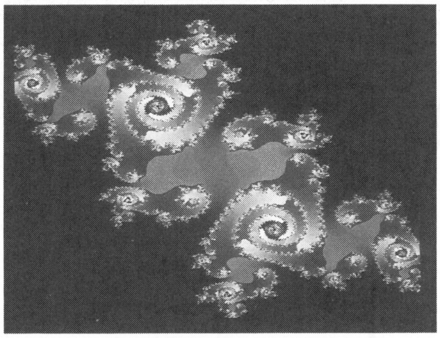
複素数を2乗して定数を足していく数列の場合，まず数列の1番目，「初項」の値をどう取るかによって，数列の振る舞いに差が出てくる。また同様に，足していく定数をどのように取るかも数列の増減に影響を与えるかもしれない。

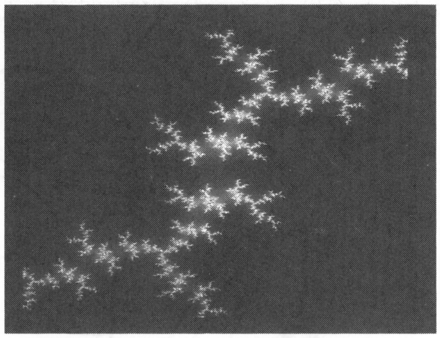
表1を見てほしい。これは，初項と加算する定数を変えた3つの数列を，それぞれ第5項まで計算した結果である。ここに挙げた3つの例の，初項と加算定数にはそれほどの違いはないのに，数列の振る舞いはバラエティーに富んでいる。特に，1番目の収束する数列と，3番目の振動する数列の振る舞いはなかなか興味深い。

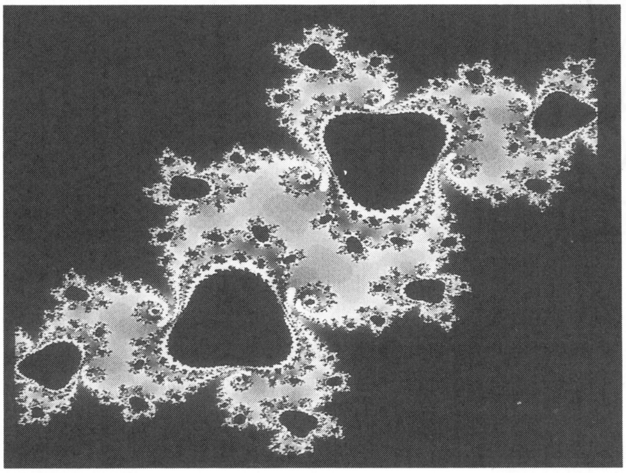
では，加算する定数を任意の値に固定して，初項の実数値と虚数値をさまざまに取るとする。ある数列は発散するだろうし，そうでない数列は収束するか，振動する。数列が「発散しない」場合には，初項の複素数に対応する複素平面に点を打っていけば，数列の振る舞いを表す「地図」が出来上がる。

要するに，初項の実数値を横軸の座標値として，虚数値を縦の座標値として取った点のうち，数列が発散しない場所に点を打っていけばいい。

ここで示したような点が集まった図形が，有名なジュリア集合である。フラクタルの解説書には必ずといっていいほど登場する，大小の渦巻が配置されたような図形のことだ。







ジュリア集合の振る舞いを予測するのは難しい。数列が急激に発散して画面が真っ黒になったかと思えば，思いがけず美しい模様を描くこともある。
適当に定数値を設定して小さい集合を描き，全体像を摑んでから，きれいなものを大きく描く，というのが，フラクタルのオーソドックスな楽しみ方だ。これが面倒くさいという人は，以下の値を参考にしていただきたい。

複素定数の実数	虚数
-0.05	-0.86
-0.25	0.74
-0.15	0.65

ジュリア集合を描く

次にいよいよ，このジュリア集合を描くプログラムを組むことを考えよう。とはいっても，BASICで虚数を扱うことはできないのに，どうやってプログラミングすればいいのか，と思うかもしれないが，心配はいらない。プログラム中，虚数項と実数項を分けて計算をすればいいのである。

また複素平面に点を打つとき，座標を取るため実数値と虚数値が必要であることを考えると，こうしたほうが便利である。

始めにプログラムの方針をざっとさらおう。一定の範囲をもつ複素平面の地図を作るのだから，プログラムは2つのループをもつ。外側のループカウンタを数列の初項の虚数値，内側のループカウンタを初項の実数値としよう。この初項から数列の計算を始めるわけだが，自分自身を2乗し，指定した定数を足していった結果，数列が発散しない点に色をつけていけばジュリア集合の出来上がりである。

まずはお決まりの，画面の初期化コマンドを書き込む。画面は，512×512ドットの，65535色モードに設定しよう。次に，プログラムのメインループとなる，2重のネスティングを作る。行番号は2000行あたりから始めるといい。ループカウンタに2つの変数が必要だから，変数の宣言をしなければならないが，この宣言は「必要になったら先頭につけ足す」ようにする。プログラムは，なにも先頭から順序よく書いていく必要はないのである。

ところが，ここで問題が生じる。複素平面にまんべんなく点を打っていくためには，初項となる複素数を小数範囲で増やしていかなければならない。ところがX-BASICのfor〜next文では，カウンタにｉnt型の変数しか使えないから，このループカウンタを直接数列の初項として使うことはできないのだ。

問題を簡単にするために，ジュリア集合を描く複素平面の範囲を，絶対値1の範囲に決めてしまう。ループカウンタは，画面上のドットに対応させることにしよう。またデバッグの効率を考えると，ジュリア集合を描くドット数は可変であるほうがいい。プログラムの冒頭に，描画するドット数を表す変数pmaxを宣言しよう。宣言の中で，変数に小さな値，15を代入しておく。

では，0から15まで増えていくループカウンタを，−1から1までの範囲に押し込めるにはどうすればいいだろうか。ループカウンタがnの値を取っている場合を考えると，−1から1までの範囲を16対nに内分する値を取ればいい。この先pmaxには違う値が入るだろうから，この変数をパラメータに使うと，

内分点＝(n−((pmax＋1)÷2))÷(pmax＋1)÷2

という公式を得る。この式の値は整数ではないので，この値を入れておくfloat型の変数を用意しなければならない，ということに気づくようになればたいしたものである。

さて，これでループカウンタに対応する虚数値と実数値が求められた。あとはこの2つの値を初項として数列を計算すればいい。calc_fractという関数に数列の計算をさせようと思うのだが，ここでは数列を計算する関数の本体を書くのはあと回しにしよう。とりあえず，初項となる2つのfloat型の変数を受け取る，という体裁だけを整えておいて，中身は，

return(65535)

とだけ書いておく。するとこの関数は，返り値が常に65535となるわけだ。

メインループの2つのカウンタを座標値に取る。先ほどの公式を使って，ループカウンタから複素平面上の点を求め，求めた2つの値を関数calc_fractに渡す。いまのままだと，この関数は一定の返り値しか渡さないのだが，そんなことにかまわずループを閉じ，最後に「end」と書き入れてプログラムを走らせてみよう。画面上に16×16の白い四角形が描かれれば，プログラムは正しく動作していることになる。つまり，プログラムのこの部分は正しく動作するのだから，もういじくる必要はないのだ。

動作確認をしたら，次はいよいよ数列を計算する関数を作り上げていく。繰り返しになるが，ポイントは実数項と虚数項を分けて計算する，という部分である。次の複素数を2乗することを考える。

$a+bi$

いうまでもないことだが，aは実数項の係数，bは虚数項の係数だ。これを展開公式を使い展開すると，

$a^2+2ab\,i+b^2i^2$

となる。ところが，i^2は−1なのだから，上の式は，

a^2-b^2+2abi

と同じである。すると複素数a＋biを2乗した複素数のうち，実数項はa^2-b^2で，虚数項は2abiということになる。

数列の最初の実数係数，虚数係数が関数のパラメータとして渡されているのだから，この2つをaとbそれぞれに代入すれば，数列の2番目の値を求めることができる。3番目の値を求めるためには，2番目の数列の値を使えばいい。ただ，ここでは複素数を2乗して定数を足すことになっていた。この定数のうち実数に当たる部分をre，虚数に当たる部分をimとし，float型の変数と

してプログラム冒頭で宣言しておこう。input文で任意の値を代入できるようにするのがスマートだが，この段階ではとりあえず宣言時にreに−0.19，imに0.66を代入しておく。

さて，数列の計算方法はこれでわかっていただけたと思うが，ではこの数列が発散したかどうかを判断するためにはどうすればいいか。これも意外に簡単である。数列の振る舞いを概観したときの表1を見てほしい。発散する2番目の例を見ると，実数項が3を超えたあたりから急速に値は大きくなっていく。

詳しい説明は避けるが，実はこの数列は実数係数と虚数係数の2乗を足したものが4を超えると，数列の値は必ず発散していくのである。つまり数列を計算していきながら，実数係数と虚数係数の2乗の和を監視して，4を超えるようなら発散したものとみなせばいい。ただ，数列が発散しない場合もあるので，256回計算をして発散しなかったら，収束または振動するものとみなそう。要するに，数列の計算を256回のループでくくればいいのである。

で，収束した数列に関しては65535を，発散した数列に関しては0を返り値とするように関数calc_fractを書けば，純白のジュリア集合を描くプログラムが出来上がる。XCを持っている人はプログラムをCにコンバートし，プログラム冒頭で宣言したpmaxに128などの値を代入して，大きなジュリア集合を描いてみよう。また，数列に足す定数をさまざまな値にすれば，いろいろな形のジュリア集合が描かれることになる。

マンデルブロ集合

フラクタルの解説書に出てくるジュリア集合の絵は，たいてい色づけされてあり，非常に魅力的である。いま作っているプログラムに色をつけるには，ほんの4，5行のリストをつけ足すだけでいい。まず関数calc_fractを，何回計算した時点で発散したか，つまり発散した時点のループカウンタを返り値とするように改造する。また256回計算を繰り返しても発散しなかった場合は，便宜上0を返そう。そして，この0から255までの返り値によって，さまざまな色をつけるのである。どのような色をつけるかは，リスト1の関数get_colorをそのまま拝借すればいい。ちなみにcalc_fractの返り値を渡すことによって，その値に対応した色を返す関数である。

表1　数列の振る舞い

初項＝1+i　加算する定数＝1−i

	第1項	第2項	第3項	第4項	第5項
実数項	1	1	1	1	1
虚数項	i	i	i	i	i

例1：同じ値に止まる（収束する）場合

初項＝1+i　加算する定数＝−i

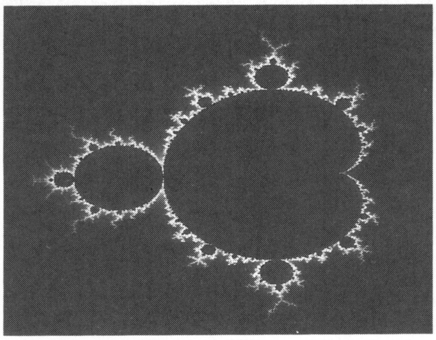
	第1項	第2項	第3項	第4項	第5項
実数項	0	−1	0	−9	80
虚数項	i	−i	−3i	−i	17i

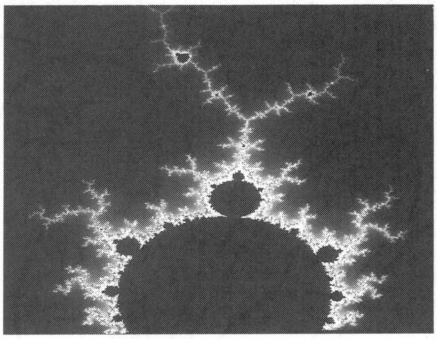
例2：絶対値が大きくなっていく（発散する）場合

初校＝1/2+i　加算する定数＝1/2

	第1項	第2項	第3項	第4項	第5項
実数項	0	−1/4	9/16	209/256	…
虚数項	i	0	0	0	…

例3：近い値をいったりきたり（振動）する場合


マンデルブロ集合の全景


どこを拡大してもマンデルブロ集合

ところで，ジュリア集合と同じくらい頻繁に，フラクタルの参考書に出てくるのがマンデルブロ集合だ。このマンデルブロ集合は，実はジュリア集合を描くプログラムにほんの少し改造を加えるだけで描くことができる。

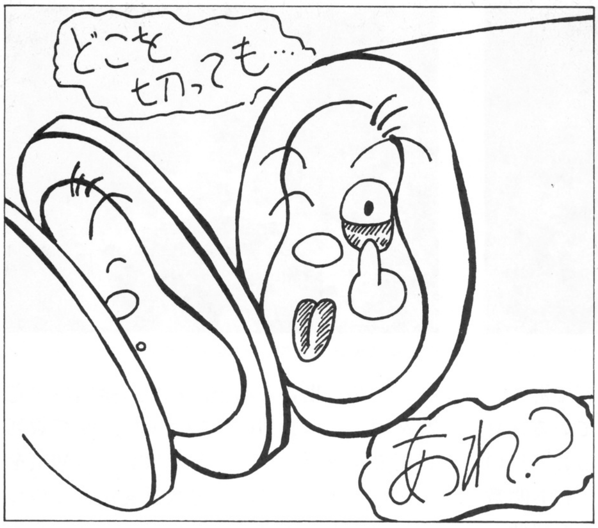
ジュリア集合の場合，数列に毎回足す定数を固定し，数列の初項を変化させて数列の振る舞いを調べた。今度は数列の初項を0として，数列に足す定数を変化させつつ数列の行方を調べてみよう。つまり，2重のメインループのループカウンタを定数に割り当てて，数列を計算する関数に渡すのだ。

ジュリア集合を描くプログラムの，関数calc_fractの名前をcalc_mandとでも変えてしまおう。で，関数に渡されたパラメータは数列の第1項として実数部と虚数部に代入されていたのだから，これを0を代入するように書き換える。

関数に渡されたパラメータはどうするかというと，これは数列に足す定数となる。するとジュリア集合の数列を計算していたときの定数の虚数部と実数部，imとreは必要なくなるから，変数宣言を含めプログラム上から消し去ってしまう。これだけの改造を施すだけで，ジュリア集合を描くプ

リスト1

```
1000 /* 自己平方フラクタル
1010 screen 1,3,1,1
1020 int i,j,k,p : float pmax = 256,phalf
1030 float f1,f2,f3,ii,jj,re,im
1040 locate 0,0
1050 input "描画する大きさ",pmax : phalf = (pmax+1) / 2
1060 input "複素定数の実数",re
1070 input "複素定数の虚数",im
1080 for i = 0 to pmax
1090  for j = 0 to pmax
1100 locate 0,3 : print j,i
1110  ii = (i-phalf)/phalf : jj = (j-phalf)/phalf
1120  p = fr_calc( ii,jj ) mod 256
1130  pset( i,j,get_color( p ) )
1140  next
1150 next
1160 end
1170 func int fr_calc( f1;float,f2;float )
1180 int i=1
1190 float a,b,c,d,e
1200  a = f1 : b = f2 : e = f1*f1+f2*f2
1210  repeat
1220   c = pow(a,2)-pow(b,2)+re
1230   b = 2*a*b+im : a = c
1240   e = a*a+b*b
1250   if e > 4 then break
1260   i = i + 1
1270  until i > 256
1280 return( i )
1290 endfunc
1300 func get_color( p )
1310  return( rgb( p mod 32,(p/2) mod 32,(p/4) mod 32 ) )
1320 endfunc
```



ログラムはマンデルブロ集合を描くプログラムに変身する。なんともあっさりした話である。

ジュリア集合の場合，数列に足す定数を変えることによって，図形の形がいろいろ変化した。ひと口にジュリア集合といっても，いろいろな形があるわけだ。しかし，ジュリア集合の場合は数列に足す定数をパラメータとして取ったのに対し，マンデルブロ集合ではそのパラメータがなくなってしまっている。誰かがマンデルブロ集合といったなら，その場合はひとつの形しか指さない。

マンデルブロ集合が興味深い振る舞いを見せるのは，集合の一部を拡大したときだ。大きな円の周囲に寄生する小さな団子のような領域を拡大すると，そこにもさらに小さな団子がくっついている。

そういった意味では，正しいマンデルブロ集合描画プログラムとは，任意の領域を拡大できなければならない。ジュリア集合を描くプログラムを改造したいまの状態では，描画範囲は絶対値1の範囲に固定されてしまっている。これを，どの範囲を描くかを指定できるように改造すればいいのである。サンプルとして掲載したプログラムが参考になるかもしれない。

フラクタルの深淵へ

実際に自分でプログラムを組んでみると，単純な作業の積み重ねから複雑な図形を描き出すこのプログラムの不思議さを実感できるはずだ。ただ，きれいな反面，描画時間がかかってしまうのが欠点である。いちばん時間を食うのは，数列が発散しない場合が多い図形だ，というのは想像に難くない。すると，一般的にジュリア集合よりマンデルブロ集合を描くほうが多くの時間を必要とする，といえる。512×512のフルサイズ描画を試みるなら，ぜひCにコンバートしたいものだ。

さて，ここで取り上げたマンデルブロ集合やジュリア集合のような図形を「フラクタル」図形というわけだが，このフラクタルという言葉の定義は案外ややこしい。「自己相似性がどうの」とか「接線が引けず微分不可能がこうの」などとはよくいわれることだが，難しいことを考えず「不思議と人を惹きつける図形」くらいに思っていただければいいだろう。

しかし，せっかくフラクタル図形を描く方法がわかるようになったのだから，「不思議だ」と眺めるばかりでなくフラクタルのもう少し深い概念に触れてみるのもいいのではないか。

たとえば，「雲」というのはどうしてできるのか，あるいはもっと進んで，どうして雲はあのような形になるのかを考えてみよう。上空の湿度や温度は雲の形を決定する要因になるかもしれないと思い，空の限られた領域を格子状に区切り，1つひとつの格子の湿度，温度の振る舞いを記録してみるのだが，どうもこの記録と雲の形状の関連がつかめない。

ここで思い出してほしいのは，ジュリア集合を描くプログラムの原理である。上空の湿度などを記録するのと同じように，ここでも数列を計算したのだが，その数列の振る舞いだけからジュリア集合の全体像を想像することはできない。すると同様に雲の形もフラクタルなのではないか，と考えるのは当然である。そしてもし雲がフラクタルなら，フラクタルを外側から概観することができれば，湿度や温度を計るのとはまったく違ったアプローチで雲の挙動を捉えることはできないものだろうか。

カオス理論という言葉を聞いたことがある人は多いと思う。ここで取り上げた話は，最近天気予報に応用されて高い的中率を上げている，「カオス思考」の出発点を模したものである。

このように，フラクタルというのはなかなかに奥の深い概念だ。しかし，マンデルブロ集合のような複雑な図形を描くことも，原理を追うだけなら難しくない。複素数とか複素平面とかいった，少々の専門知識(とはいっても高校で教わる程度の数学だが)を理解しさえすれば，初心者にもプログラムが作れてしまうものなのだ。

ところで，ここまで大げさに「初心者でも作れます」的な記事を載せてしまうと，読者の投稿ネタの範囲を狭める結果になりはしないかと少々不安になる。まあ，それでも読者のレベルアップは歓迎すべきことではある。。

参考文献
フラクタルとは何か，小川泰著　岩波書店刊

リスト2

```
1000 /* マンデルブロ集合
1010 screen 1,3,1,1
1020 int i,j,k,p : float pmax = 256,phalf
1030 float f1,f2,f3,ii,jj,rs,is,re,ie,rd,id
1040 locate 0,0
1050 input "描画する大きさ",pmax : phalf = pmax / 2
1060 input "複素平面の実数値の始点",rs
1070 input "複素平面の虚数値の始点",is
1080 input "複素平面の描画範囲",rd
1090 for i = 0 to pmax
1100  for j = 0 to pmax
1110 locate 0,5 : print j,i
1120  ii = i*rd/pmax+rs : jj = j*rd/pmax+is
1130  p = mn_calc( ii,jj ) mod 256
1140  pset( i,j,get_color( p ) )
1150  next
1160 next
1170 end
1180 func int mn_calc( f1;float,f2;float )
1190 int i=0
1200 float a,b,c,d,e
1210  a = 0 : b = 0
1220  repeat
1230   c = pow(a,2)-pow(b,2)+f1
1240   b = 2*a*b+f2 : a = c
1250   e = a*a+b*b
1260   if e > 4 then break
1270   i = i + 1
1280  until i > 255
1290 return( i )
1300 endfunc
1310 func get_color( p )
1320  return( rgb( p mod 32,(p/2) mod 32,(p/4) mod 32 ) )
1330 endfunc
```

雪景色を楽しむための

お手軽降雪シミュレータ

Tan Akihiko 丹 明彦

適当に描いたグラフィックに雪を降らせる。そんなかわいいプログラムをX-BASICで書いてみました。ゆっくりコタツにでも入りながら，好きなだけ雪を降らせてみるのもいいかもしれませんね。

絵心がないとお嘆きの諸兄に贈る自然物景観シミュレータへの1プロジェクトとして，降雪シミュレータをでっちあげてみた。基本的にはありふれたアルゴリズムであるが，野心的な方向性も示唆する。

基本戦略

降り積もる雪を表現するためにはいくつかの戦略が考えられるが，今回は雪をブラウン運動を行う粒子としてモデル化してみた。要するに乱数を使って自動生成するということだ。

地形や雲，植物などのように，形に法則性のようなものがあるのに規則的でないものには，上手に乱数を導入するという手軽で強力な手法が存在する。雪もこの方法で結構それらしくなる。

今回のプログラム

ペイントツールなどで描いた絵に後処理で雪を生成する。約束事は3つ。

1) 65536色モードで描くこと
2) 真横から見た絵を描くこと
3) 空間部分を黒で描くこと

これを“SRC.PIC”というファイル名でセーブしておくこと（もしくはプログラム中で指定しているファイル名を変更すること）。

プログラム本体はX-BASICで動作する。画像ファイルとしてAPICフォーマットを用いる場合はAPIC.FNCが必要である。

プログラムをRUNコマンドで実行すればあとは待つだけである。適当に降り積もった頃にジョイスティックのトリガを押せば，空中に舞う雪を適当にあしらって終了する。ただし，とても時間がかかる。目安としてはX68030でもひと晩かかる。コンパイルすれば同じことが数分でできるので，C言語の環境を揃えられる人はコンパイルして使うことをお勧めする。

2次元版のアルゴリズム

雪の粒子を画面最上ラインで発生させ，乱数で揺らしながら下に落とし，着地したところで固定する。この着地判定にドットの色を用いている。黒でないドットにぶつかったら止めるわけである。

ただ，これだけでは雪が降り積もらないで無数の霜柱が立ってしまう。雪は雪の上にも重なるので当然なのだが，霜柱は避けねばならない。

そこで，下へ降りられるだけ降りるようにする。といっても簡単で，乱数で揺らすことを3回繰り返すだけ。3回チェックして1回でもさらに降りられそうならその方向に降りる。3回とも着地するなら着地する。たったこれだけのことでずいぶんそれらしくなる。

あと，雪粒が四角いドットでは物足りないので，白い雪粒の周囲に灰色のドットを置いてなんとなくアンチエリアシングっぽくしてある。この灰色のドットはすでになにかが描画されている場所には描画しない。簡単な木と家を描いて実行してみると，木の枝のくぼみの部分にはたくさん雪が溜まり，家の軒下にも少しずつ雪が積もっている様子がわかる。まあまあの効果ではなかろうか。

このプログラムの最大の問題点は，下絵を描く際にそれなりの絵心が必要なことであろう。

3次元版のアルゴリズム

今回は掲載を見送っているが，3次元空間内に雪を降らせるプログラムも試しに作

図1　単純な着地判定

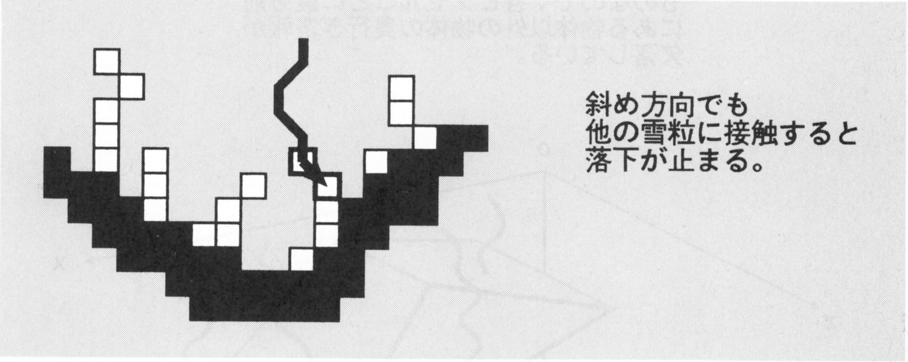
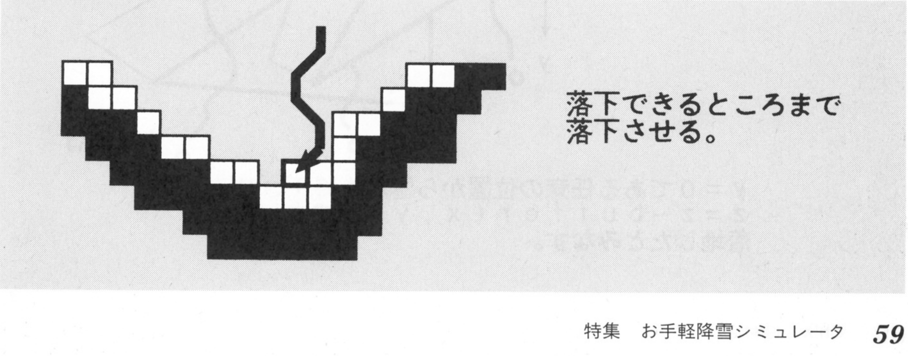


図2　改良された着地判定





この元絵に雪を降らせてみると……



なんとなくそれっぽく見える

ってみた。結構いけそうだ。Ｚバッファアルゴリズムで描画した画像をベースに後処理を行うものである。

基本的な処理は同じ。画像の上端で発生させた雪の粒子をX(横方向)とZ(奥行き方向)の両方で揺らしてY(下方向)に落とし，着地した場所に描画する。この着地判定が工夫のしどころなのだ。

奥行きの表現としてＺバッファを用いたのは，データ量または処理時間の爆発的な増加を抑えるためである。２次元版を単純に拡張すると，ボリュームデータを用意する必要があり，これは数メガバイトしかメモリをもたないX68000/030ではきつい。また，まじめに粒子をシミュレートすると，雪の粒子を無数の３次元オブジェクトとして表現する必要がある。これらを相互に接触判定をやっていたのでは，計算時間がいくらあっても足りない。Ｚバッファは，データ量と計算時間の両方でリーズナブルな選択である。

もちろん欠点もある。Ｚバッファアルゴリズムを用いて生成した画像は，３次元の情報を完全にもっているわけではない。ほかの物体のうしろに隠れている物体の奥行き情報が欠落している。場合によっては正しく積雪をシミュレートできない。さらに，視線がなるべく水平になるようにレンダリングしておかないと，重力は画面の縦方向に働いていることになっているので，妙な結果になってしまう。

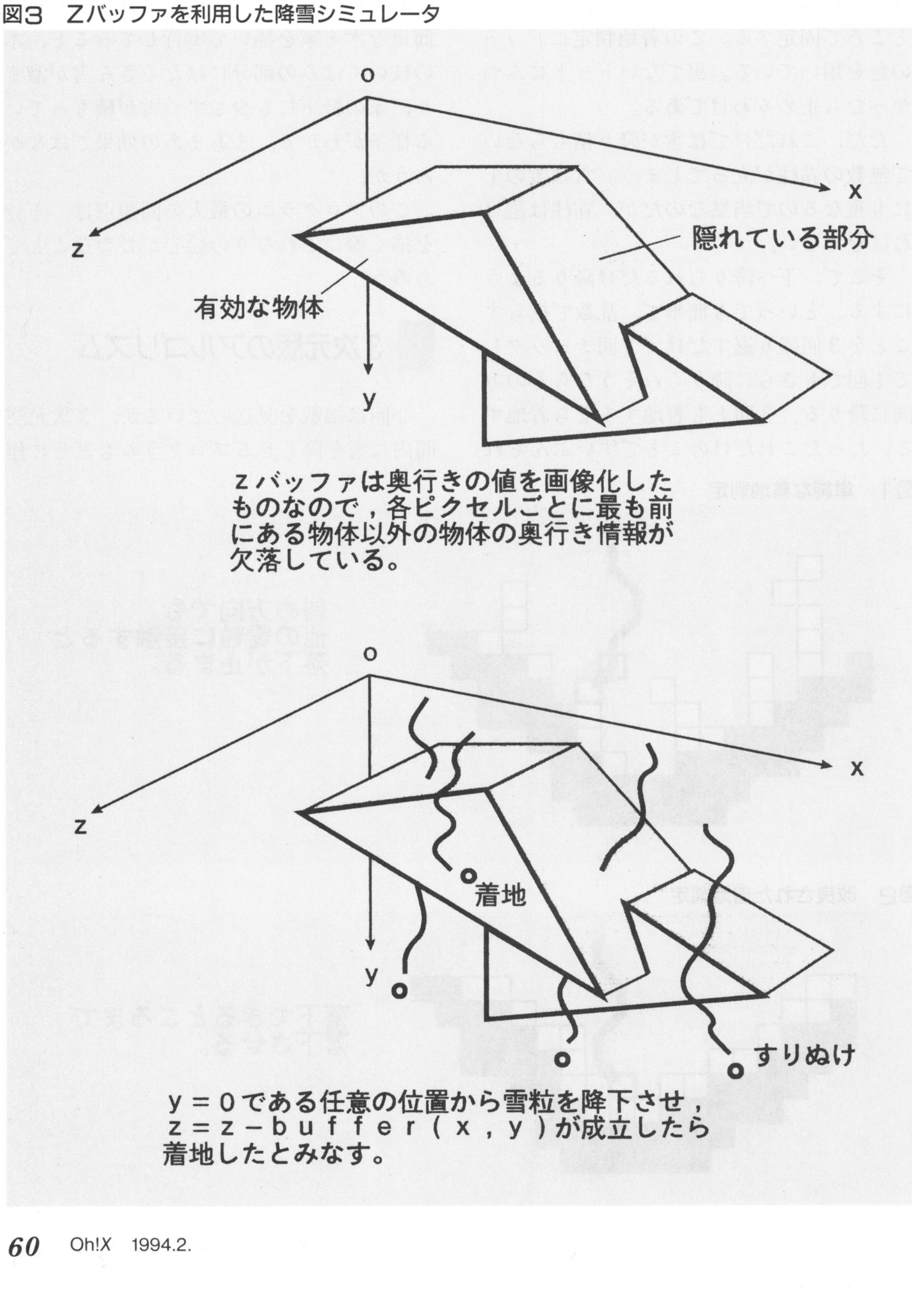
つまり，この３次元版積雪シミュレータは欠点を理解したうえで使う必要があるのだ。３次元コンピュータグラフィックはマシンパワーで押し切るか，ごまかしで稼ぐかで実用的な速度を出す技術なので，この積雪シミュレータも正しく３次元コンピュータグラフィックしているといえる。

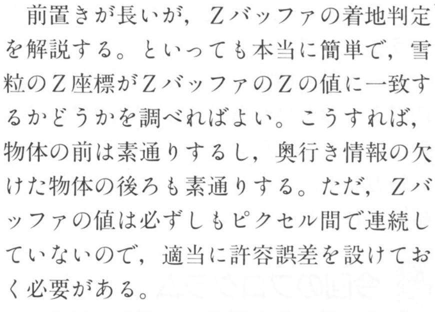
前置きが長いが，Ｚバッファの着地判定を解説する。といっても本当に簡単で，雪粒のＺ座標がＺバッファのＺの値に一致するかどうかを調べればよい。こうすれば，物体の前は素通りするし，奥行き情報の欠けた物体の後ろも素通りする。ただ，Ｚバッファの値は必ずしもピクセル間で連続していないので，適当に許容誤差を設けておく必要がある。

これは，実際に４年以上前に作ったＺバッファレンダリングプログラムを引っ張り出して実験してみた。地面のポリゴンの上に字の形をしたポリゴンを浮かせたシーンをＺバッファアルゴリズムでレンダリングして，結果の画像とＺバッファをセーブしておき，積雪シミュレータを実行する。

ポリゴンの陰になっている部分は雪の積もりかたが少なくなっており，これもまあまあの結果といえるのではないかと思う。

図３　Ｚバッファを利用した降雪シミュレータ





3D版の実験画像

野望と展望

今回は諸事情から２次元版のみの掲載となる。３次元版を実現するためには，ある程度ちゃんとしたＺバッファレンダリングプログラムが必要だし，Ｃ言語によるプログラミングも事実上必須になる。そうなればもう大作なので見送ってしまった。

なお，３次元版とはいっても，２次元画像＋Ｚバッファであるから，上手に実装すればZ's-EXなどに組み込むことも可能かもしれない。実は今回の３次元版のアイデアは，ペイントツールにＺバッファを追加してマウスから奥行きを簡単に入力できる仕掛けを考えれば３次元のペイントシステムができるという発想からきている。モデリングやレンダリングの不要な３次元コンピュータグラフィックが実現できるわけだ。ちゃんとした3Dではないから，くるくる回したりはできないが，できるだけ少ないコストでそこそこの効果を得るというコンピュータグラフィックの鉄則には見事に沿っているのである。

たまにはアイデア一発のお気楽なプログラムもいいものである。

雪のアニメーション

今回のプログラムは，降雪シミュレータというよりも積雪シミュレータという感じで，雪が降っているところは見えない。やはり雪が動いていないと面白くない。秋川氏が最近トースター（AMIGAのVideo Toaster 4000）を買ったというので見せてもらったら，雪が降るようなビデオエフェクトがあって，これが結構気持ちよかった。AMIGAの卓越したアニメーション機能が使われているのだろうとは思うが，それっぽい動きには見習うべきものがある。

数百の雪の粒子をリアルタイムで動かして，なおかつ積もらせ，さらに３次元化できれば，見栄えのするデモプログラム（またはスクリーンセーバー）になる気がする。

今回のプログラムは，雪が積もるにつれて重みがかかって屋根の端付近から滑り落ちるという部分までシミュレートする壮大な計画もあったのだが，そこまでやらなくてもそこそこの品質が得られたのでよしとした。が，上のようなデモプログラムであれば，滑り落ちる雪とか，雪の重みにしなる木の枝とか，そういった風流なフィーチャーもあると嬉しいに違いない。あとは時刻によって背景の明るさが変わるとか，陽が当たると溶けるとか，子供が出てきて雪だるまを作り始めるとか……。だんだん嘘っぽくなってきたなあ。

あと，今回のプログラムは，特にインタプリタでは暴走したかと思うほど遅いので，せめて配列statの初期化中は進行状況が見えるようにドットを表示している。



リスト１

```
 10 char tmp(1023), stat(511,511)
 20 int i, j
 30 screen 1,3,1,1
 40 apic_load("src.pic")    /* 元画像ロード，img_load()なども可。
 50 for i = 0 to 511
 60   if (i mod 8)=0 then print ".";
 70   get( 0, i, 511, i, tmp )
 80   for j = 0 to 511
 90     if (tmp(j*2)+tmp(j*2+1))<>0 then stat(i,j)=1 else stat(i,j)=0
100   next
110 next
120 int WHITE, GRAY
130 int ok, x, y, dx
140 WHITE = &HFFFE: GRAY = rgb(16,16,16)
150 while 1
160   /* ジョイスティックのボタンで終了
170   if strig(1)<>0 then break
180   x = (rand() mod 510) + 1
190   for y = 0 to 509
200     ok = 0
210     for i = 1 to 3    /* なるべく下へいくようにリトライ
220       dx = (rand() mod 3) - 1
230       if x+dx > 511 then ok = 1: break
240       if x+dx < 0 then ok = 1: break
250       if ( stat(y+1,x+dx) = 0 ) then ok = 1: break
260     next
270     if ok = 0 then break      /* 着地
280   x = x + dx
290   next
300   /* 積もる雪を表示
310   if x < 1 then continue
320   if x > 510 then continue
330   i = y*512+x
340   pset(x,y,WHITE): stat(y,x)=1
350   if stat(y-1,x) = 0 then pset(x,y-1,GRAY)
360   if stat(y,x-1) = 0 then pset(x-1,y,GRAY)
370   if stat(y,x+1) = 0 then pset(x+1,y,GRAY)
380   /*if stat(y+1,x) = 0 then pset(x,y+1,GRAY)
390 endwhile
400 /* 降る雪を表示
410 for i = 1 to 200
420   x = (rand() mod 510)+1
430   y = (rand() mod 510)+1
440   if stat(y,x) = 0 then {
450     pset(x,y,WHITE)
460     if stat(y-1,x) = 0 then pset(x,y-1,GRAY)
470     if stat(y,x-1) = 0 then pset(x-1,y,GRAY)
480     if stat(y,x+1) = 0 then pset(x+1,y,GRAY)
490     if stat(y+1,x) = 0 then pset(x,y+1,GRAY)
500   } else continue
510 next
520 end
```

これがフラクタル圧縮だ(嘘)

ヒルベルト曲線を利用した画像圧縮の試み

Tan Akihiko 丹 明彦

降雪シミュレータをお贈りした丹氏の次なる作品は，フラクタル図形のひとつであるヒルベルト曲線の性質を使った画像圧縮プログラムです。ランレングスパスをヒルベルト曲線にして圧縮効率が上がるのか，を考えます。

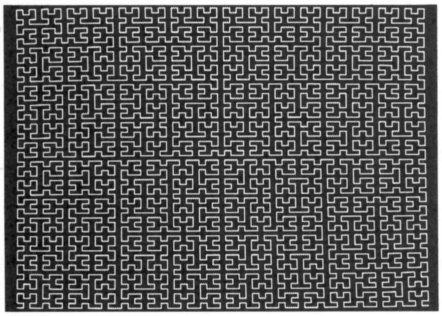
構想3年の圧縮方式をでっちあげる。ランレングス法の変形だが，もしかすると2次元画像に強いかもしれない。

ヒルベルト曲線

ヒルベルト曲線はいわゆるフラクタル図形のひとつである。写真のような網目模様は，実はひとつながりの線なのである。

ヒルベルト曲線の生成法の基本を図1に示す。初めはひとつのコの字形である。その辺をそれぞれ折り曲げる。その出来上がりの図形もまたコの字形を含んでいる。これをさらに再帰的に折り曲げていくと完成する(図2)。

リスト1にヒルベルト曲線を描くプログラムHILBERT.BASを示す。X-BASICのプログラムで，インタプリタでもまあまあの速度で描けるはず。プログラムそのものは再帰呼び出しを多用したものである。コの字形を時計回りに描く関数cw()と反時計回りに描く関数ccw()とは，さらにサイズを小さくしてコの字形をいくつか描く。これをタートルグラフィックふうに実現している。引数のlevelは再帰レベルで，この値が大きいほど再帰が深く，ヒルベルト曲線が複雑になる。ちなみにX68000の画面の場合，再帰レベルに与えて意味があるのは9までである。



これがヒルベルト曲線の基本形

圧縮への応用

さて，ヒルベルト曲線は，見てわかるとおり，存在範囲が矩形であり，その中での線の密度が一様である。たとえば，HILBERT.BASにおいて再帰レベルを9，1本の線分の長さLengthを1にすれば(コメントアウトしてあるのを取ればよい)，画面の全画素を1回ずつアクセスするようになる。しかも，ヒルベルト曲線は，複雑そうに見えるが，ひとつながりの線である。

そこで，圧縮に応用することを考える。ヒルベルト曲線に沿って画素の色を調べ，いくつ同じ色が続いたかを記録する。つまりランレングス圧縮の変形版である。

ランレングス法は，「隣り合う画素の色は似ていることが多い」という画像の性質(コヒーレンシー)を利用した圧縮方法である。ランレングス法では画素の色をラスタ順に調べ，いくつ同じ色が続いたか(これがrun-length)をファイルなどに記録する。これは多くのマシンにおけるグラフィックメモリの構造からそうなっているのだろう。ただ，この方法だと，左右両隣りの画素の色が似ていることは利用できても，上下は利用できない。画像は2次元なのだから，なんとなくもったいない。

そこで，画面を網目のようにまんべんなくアクセスするヒルベルト曲線を用いることにする。これを使えば，上下左右のコヒーレンシーを活用できそうである。

ヒルベルト曲線を用いた圧縮

リスト2にヒルベルト曲線を用いた圧縮プログラムHILCOMP.BASを示す。リスト2の51行以降はリスト1の25～44行を流用するように。また，掲載されている状態ではHILCOMP.BASは圧縮プログラムであるが，30行のmの値を1にすれば展開プログラムにもなる。

実行速度が遅いので事実上コンパイルして使うことが必要である。プログラムの性

図1 基本パターン

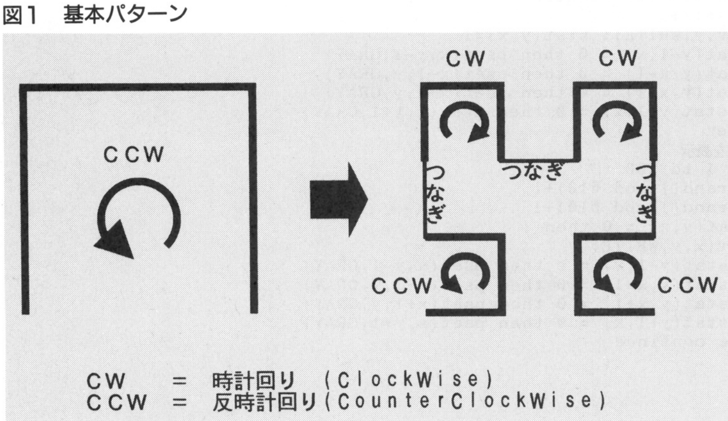
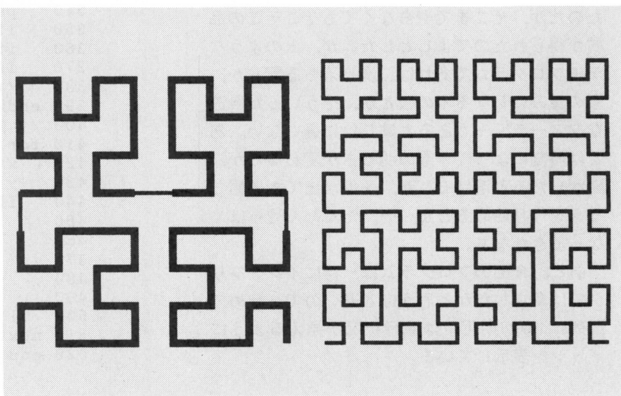
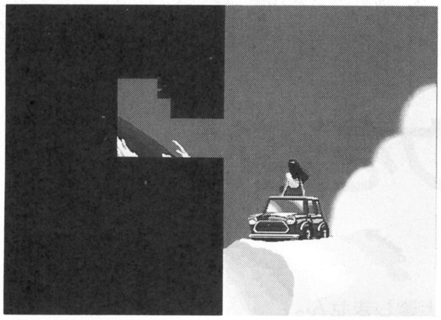


図2 再帰的に細かくなる様子





画像展開中の画面

質上グラフィックの初期化は困るので，BC.XでC言語にコンバートしたあと，例のb_init()とb_exit(0)の行を削除してからコンパイルするようにしてもらいたい。

圧縮プログラムは，65536色モードで画像を表示した状態で起動すること。ファイルネームを聞いてくるので，作りたい圧縮ファイルの名前を入力する。コンパイルしても死ぬほど遅いので，暴走したかと思わないこと。やっぱりCで書いてスーパーバイザモードでガシガシやらなくてはだめだ。

展開プログラムはいきなりファイルネームを聞いてくるので，圧縮プログラムで作成したファイル名を入力すれば，そのファイルを読み込んで展開し，復元画像を表示する。

圧縮ファイルフォーマット

ファイルフォーマットはなにも考えられていない。ファイルの先頭から，色（2バイト）とランレングス（3バイト）をひと組みにしてひたすら並んでいる。

ランレングスが3バイトというのはかなり冗長だが，X68000の65536色モードは約25万画素あるので，2バイトに入りきらない可能性を考慮した。

符号化はやらないと話にならないだろう。ランレングスの分布には明らかな偏りがあるので，それに応じた符号表を作れるはずだ。

結果

結論を先にいうが，この方式はそれほどおいしくない。

私はこの圧縮プログラムをまずC言語で制作してX-BASICにコンバートした。C言語バージョンを開発中に，通常のランレングス法の圧縮プログラムも比較のために作っておいた。それと比較したところ，画像によっては通常のランレングス法のほうが効率よく圧縮できていた。

ヒルベルト曲線を用いたこの方法には，グラフィックメモリをリニアに使えないという大きな欠点がある。このデメリットを補うほど圧縮効率は期待できないこともあり，あまり使えそうもない。

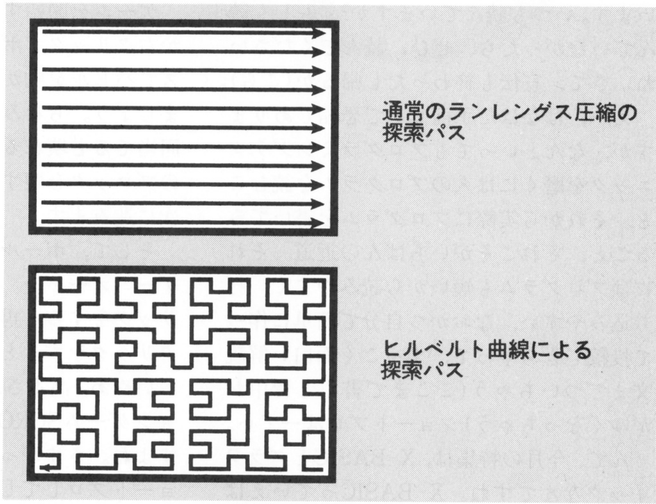
終わりに

3年前にヒルベルト曲線の存在を知ってから，ひょっとすると結構イケル圧縮プログラムではないかと思った私の目論見は見事に外れた。

ただ，圧縮画像ファイルを読み込んで表示するところは少し面白いので，圧縮プログラムでなく画像エフェクタとしてなら生き残る道があるかもしれない。

なお，世の中にはフラクタル圧縮というものが存在するそうだが，この圧縮プログラムとは関係はない。

図3　ランレングス圧縮の探索パスをヒルベルト曲線にする



リスト1

```
/* ヒルベルト曲線
int tmp, Length, Dx, Dy, x, y, ox, oy
/* メインルーチン
screen 1,3,1,1
x = 511: y = 511: ox = 511: oy = 511
Length = 8: hilbert( 6 )
/*Length = 1: hilbert( 9 )    /* 全面埋めつくし
end
/* タートルグラフィックスもどき
func goForward()
  ox = x: x = x + Dx
  oy = y: y = y + Dy
  line( ox, oy, x, y, 65535, 65535 )
endfunc
func setUp()
  Dx = 0: Dy = -Length
endfunc
func turnRight()
  tmp = Dx: Dx = -Dy: Dy = tmp
endfunc
func turnLeft()
  tmp = Dx: Dx = Dy: Dy = -tmp
endfunc
/* ヒルベルト曲線下請け
func cw( level;int )    /* 時計回り(clockwise)
  if level>0 then {
    turnLeft(): ccw( level-1 ): goForward()
    turnRight(): cw( level-1 ): goForward()
    cw( level-1 ): turnRight(): goForward()
    ccw( level-1 ): turnLeft()
  }
endfunc
func ccw( level;int )    /* 反時計回り(counterclockwise)
  if level>0 then {
    turnRight(): cw( level-1 ): goForward()
    turnLeft(): ccw( level-1 ): goForward()
    ccw( level-1 ): turnLeft(): goForward()
    cw( level-1 ): turnRight()
  }
endfunc
/* ヒルベルト曲線
func hilbert( level;int )
  setUp(): turnLeft(): ccw( level )
endfunc
```

リスト2

```
/* ヒルベルト曲線を利用した画像圧縮(試作)
int _SAVE = 0, _LOAD = 1
int m = 0    /* _SAVE
int tmp, Length, Dx, Dy, x, y, ox, oy
int fp, rle, c, cl, hb, mb, lb
str filename
/* メインルーチン
input "filename: ",filename
if m = _SAVE then fp = fopen( filename, "c" )
if m = _LOAD then screen 1,3,1,1: fp = fopen( filename, "r" )
x = 511: y = 511: ox = 511: oy = 511
Length = 1    /* 全面埋めつくし
if m = _SAVE then c = point(x,y): rle = 1: hilbert( 9 ): saveSeg()
if m = _LOAD then loadSeg(): pset( x, y, c ): hilbert( 9 )
fclose( fp )
end
/* カラーコードとランレングスのファイル読み書き
func saveSeg()
  fputc( c/256, fp )
  fputc( c mod 256, fp )
  fputc( rle/65536, fp )
  fputc( (rle/256) mod 256, fp )
  fputc( rle mod 256, fp )
endfunc
func loadSeg()
  hb = fgetc(fp)
  lb = fgetc(fp)
  c = (hb shl 8)+lb
  hb = fgetc(fp)
  mb = fgetc(fp)
  lb = fgetc(fp)
  rle = (hb shl 16)+(mb shl 8)+lb
endfunc
/* タートルグラフィックスもどき
func goForward()
  ox = x: x = x + Dx
  oy = y: y = y + Dy
  if m = _SAVE then {
    cl = point( x, y )
    if c <> cl then {
      saveSeg()
      c = cl: rle = 1
    } else rle = rle + 1
  } else {                  /* m = _LOAD
    rle = rle - 1
    if rle = 0 then loadSeg()
    pset( x, y, c )
  }
endfunc
/* これ以降はhilbert.basと共通
```

X-BASICで3Dブロック崩し
ショートプロのテクニックを盗め

Komura Satoshi 古村 聡

掲載されたプログラムを打ち込むだけでは，プログラミングは上達しません。打ち込みながらでも，遊びながらでもプログラムを読みこなすことが大切なのです。他人のテクニックを盗み，自分のものにしましょう。

どもども一っ。ショートプロの(で)でございます。毎月，「(で)のショートプロぱーてぃ」という連載で読者の皆さんから送られてきたショートプログラムを紹介しています。いつも読んでいますか? もしも読んでいなかったら，ぜひ，読んでくださいね。さて，宣伝も終わったし帰るか(こら)。

えー，はなはだ手前ミソで恐縮でありますが，なんといってもプログラミングテクニックを磨くには人のプログラムを読むこと。それから実際にプログラムを書いてみること，それこそがいちばんの近道。それにはプログラムも短いから読みやすく，打ち込みやすい，なおかつ自分で簡単に作って投稿できちゃって，すっごく面白い解説文までついちゃう(ここまで書くと背中がかゆくなっちゃう)ショートプロぱーてぃ。

んで，今月の特集は，X-BASICとグラフィックなんですね。X-BASICっていえばショートプロ。グラフィックといえば……えーっとゲームですよね(ちょっと強引)。ということでショートプロに送られてきたゲームプログラムを打ち込み，遊んで，さらにプログラムを読んでテクニックを盗んでしまいましょう。

これが題材だ

さてさてそれではさっそくプログラムを紹介しましょう。どうぞっ。

BLOCK3D.BAS for X680x0 (X-BASIC)
宮城県 高木大輔

えーっ，プログラム名そのまんまなんですけど，今回紹介するプログラムはBASICで書かれたリアルで超カッコイイ3D表示のブロック崩しゲームなのです。X680x0の電源を入れて，

A>BASIC

でX-BASICを立ち上げてからリスト1を入力してください。

入力が終わったら，RUNでプログラムを実行すると四角いチューブのようなものの奥にブロックが見えます。パドルをマウスで操ってブロックにボールを当て，すべてのブロックを消してください。

ゲームを開始するには，まず左クリックか右クリックでボールを発射します。マウスでパドルを動かして，ボールを打ち返しましょう。ちなみにブロックはボールを3回当てると壊れるようになっていて，全部のブロックを壊すとステージクリアとなり，次に進みます。

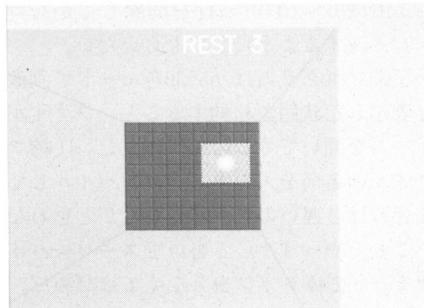
そして，ボールを3回そらしてしまうとゲームオーバー。そのときにマウスの左クリックをすると再びゲームをスタート，右クリックをするとゲームを終了します。

いやあ，高木さんは以前にも3DのアクションゲームSCROLL3Dが掲載されたんですよね。さすがっ，てすっかりのんきにショートプロしてしまった。

では，特集らしくこのプログラムの奥までつついてみましょうか。まず，このBLOCK3D.BASを遊んでみると，どこをとってもBASICとは思えないようなかっこいいゲームプログラムなんですよね。X-BASICでこれだけの3D表示をやっているのは，なかなかすごいことです。本当に表示関係を使いこなしています。

皆さんご承知のとおり，X68000にはいろいろな画面表示関係のハード，っていうか機能があります。文字を表示するテキスト，絵を表示するグラフィック，それから小さな絵と小さな背景を表示するスプライトとBGなどね。

それぞれの機能の具体的な解説は，マニュアルやほかの人に譲るとして，このプログラムにはいろんなものが表示されています。まずは，画面上に「REST」の文字，パドルとボール，ブロック，それから背景の壁が表示されています。さてここで問題です。果たして，どれになんの機能を使っているかわかりますか? 制限時間は1分。



BLOCK3D. BAS

……チッチッチ，ポーン（1分経過）

それでは答えを見てみましょう。まずは，「REST」という文字。これはテキストですよね，文字ですもん。「REST」という文字が出ている状態でBREAKキーを押してみましょう。それからカーソルを文字の上にもっていきます。テキストだったら，スペースキーを叩いていったら文字が消えていくんですよね（パチパチ，スペースキーを叩く音)。うん，見事に消えました。予想どおり「REST」はテキストでした。

次はスプライトとBGを見てみましょうか。まず，BGでできているものを消してみます。BGを消すにはと……あ，BG_SET()関数を使えばいいんですね。ダイレクトに以下のコマンドを入力してみてください。

```
BG_SET(0,0,0)
BG_SET(1,0,0)
```

BG_SET()関数っていうのは最初の引数でバックグラウンド，2番目でテキストページの割り当て指定を，そして3番目の引数を0にするとそのバックグラウンドの表示をOFFにするものです。

さて，打ち込んで……リターンキー。おおっ，奥のブロックだけが消えましたね。なるほど，ブロックだけがBGだったんですね。

続いてはスプライトをチェック。sp_init()で消えるものは……お，今度は，ボールとパドルが消えました。ということは，い

ま画面上に残っている背景はグラフィック，ということになります。あなたはいくつ当たりましたか?

このように，コーディングだけにこだわらず，このグラフィック関係の使い分けというのもとても大事なことなんですよ。特にX68000では。

以上のように，ゲームの画面構成を知りたいだけなら，プログラムを1行も読まなくてもすむんです。ほかにもスプライトパターンを知りたかったら，スプライトエディタでパターンを見ることもできますし。こんなふうにして，プログラムをちょっとずつ裸にしていくと，意外なところに意外な機能が使われているのを発見できて，とても面白いですよ。

PCGデータを2行で作る

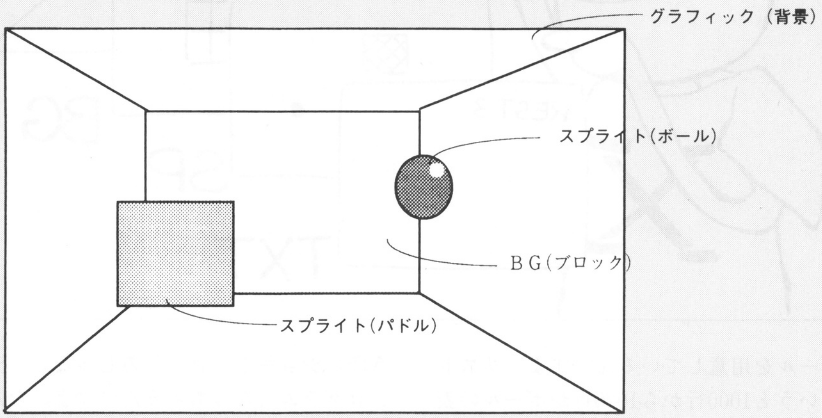
以上で，画面構成はわかりました。ボールとパドルはスプライトに描かれているということもわかりました。しかし，なんかちょっと変だと思いませんか。次なる疑問は，そう，ボールです。

このゲームでボールは奥から手前へ，手前から奥へとポンポン跳ねるんですけど，そのときにボールの大きさが変わっていますよね。X68000のスプライトには，拡大縮小の機能がないのにどうやってこんなことをしているんでしょう。

では，さっそく調べてみなくちゃ。えーっと，スプライトエディタは……と。付属のDEFSPTOOLを立ち上げてみてください。もちろん，SM.Xでもいいですけど。

なーんと，わかりますか? 画面写真を見てください。ほら，いくつも大きさの異な

図1 BLOCK3D.BASでのグラフィック，スプライト，BGの使い分け



リスト1

```
10 /*
20 /*  BLOCK3D.BAS  for X680x0
30 /*                         1993.11 D.Takagi
40 screen 1,2,1,1:contrast(0)
50 /*
60 float bx,by,bz,dx,dy,dz,zz,f(300)
70 int x,y,xx,yy,x1,y1,x2,y2,c
80 int mx,my,ml,mr,i,j,pa,bk,st,pr=3,bl=3,dt=1
90 char buf(255),blk(9,9):str t
100 for i=0 to 300:f(i)=100#/(100+i):next
110 /*
120 m_init():for i=1 to 3:m_alloc(i,20):next
130 for i=1 to 3:m_trk(i,"v15@57c"):next
140 /*
150 sp_init()
160 for i=0 to 1:for j=0 to 1:if i=0 and j=0 then continue
170   x=i*8:y=j*8:c=10-(i*2+j)*3
180   box(x  ,y,x+7,y+7,c+4)
190   box(x+1,y,x+7,y+6,c)
200   fill(x+1,y+1,x+6,y+6,c+2)
210 next:next
220 for i=0 to 14:line(i mod 2+16,i,30,i,2,&HAAAA):next
230 for i=0 to 29:r=70/(i+10):for j=0 to r-1
240   circle(i*16+39,7,r-j,15-15*j/r,,,280)
250   paint( i*16+39,7,   15-15*j/r)
260 next:next
270 for i=0 to 31
280   get(i*16,0,i*16+15,15,buf):sp_def(i,buf)
290 next
300 /*
310 for i=1 to 15
320   palet(i,hsv(90,4,i+15))
330   sp_color(i,hsv(90,5,31-i),1)
340   sp_color(i,hsv(12,15,20-i),2)
350 next
360 /*
370 img_scrn(0,1,1):window(0,0,511,511)
380 msarea(13,13,242,242):mouse(2):mouse(4)
390 /*
400 for i=0 to 1:for j=0 to 1
410   sp_move(i*2+j,114+j*14,114+i*14,1)
420 next:next
430 fill(0,0,511,511,10):fill(252,252,507,507,0)
440 for i=0 to 3
450   box(251-i,251-i,508+i,508+i,8+i)
460 next
470 apage(3)
480 fill(0,0,511,511,8):fill(256,256,319,319,4)
490 box(256,256,319,319,5)
500 /*
510 t=time$:repeat:until t<>time$
520 t=time$:repeat:block():dt=dt+1:until t<>time$:dt=dt-1
530 /*
540 bg_set(0,0,1):sp_disp(1):contrast(15)
550 repeat
560   cls:bl=3:st=-1
570   while bl
580     dx=0:dy=0:dz=0:bz=0:bk=0:st=st+1
590     for i=0 to 9:for j=0 to 9
600       bg_put(0,j+54,i+54,515):blk(i,j)=3
610     next:next
620     while bk<300 and bl:block():endwhile
630   endwhile
640   locate 10,1:print "GAME    OVER"
650   locate 11,2:print using"SCORE ####";st*300+bk
660   repeat:msstat(mx,my,ml,mr):until ml or mr
670   repeat:msstat(mx,my,ml,mr):until ml=0
680 until mr
690 mouse(0):end
700 /*
710 func block()
720   vpage(pa*2+11):pa=(pa+1) mod 2:apage(pa+1)
730   bg_scroll(0,304+x/3.2#,304+y/3.2#)
740   home(0,124+x,124+y):home(3,128+x/4,128+y/4)
750   wall(xx,yy,0):xx=x:yy=y:mspos(x,y):wall(x,y,6)
760   /*
770   if dz=0 then {
780     locate 13,2:print"REST";bl
790     bx=x:by=y:msstat(mx,my,ml,mr)
800     if ml or mr then dz=80#/dt:cls
810   } else {
820     zz=bz:bx=bx+dx:by=by+dy:bz=bz+dz
830     if bx<7 or bx>247 then m_play(1):bx=bx-dx:dx=-dx
840     if by<7 or by>247 then m_play(2):by=by-dy:dy=-dy
850     if (bz-220)*(zz-220)<=0 then {
860       x1=bx/25:y1=by/25
870       if blk(x1,y1)>0 then {
880         m_play(3):blk(x1,y1)=blk(x1,y1)-1:bk=bk+1
890         bg_put(0,x1+54,y1+54,512+blk(x1,y1))
900         bz=bz-dz:dz=-dz
910       } else if bz<220 then pr=3 else pr=1
920     } else if bz<0 then {
930       if abs(bx-x)<18 and abs(by-y)<18 then {
940         m_play(3):bz=bz-dz:dz=-dz
950         dx=(bx-x)*6#/dt:dy=(by-y)*6#/dt
960       } else dz=0:bz=0:bl=bl-1
970     } else if bz>299 then m_play(3):bz=bz-dz:dz=-dz*1.01#
980   }
990   /*
1000  x2=137+(bx-x)*f(bz):y2=137+(by-y)*f(bz)
1010  if x2<0 or y2<0 then x2=0:y2=0
1020  sp_set(4,x2,y2,bz/10+259,pr)
1030 endfunc
1040 /*
1050 func wall(wx,wy,c)
1060   int x1,y1,x2,y2
1070   x1=128-wx  :y1=128-wy
1080   x2=128-wx/4:y2=128-wy/4
1090   /*
1100   line(x1    ,y1    ,x2    ,y2    ,c)
1110   line(x1    ,y1+255,x2    ,y2+ 63,c)
1120   line(x1+255,y1    ,x2+ 63,y2    ,c)
1130   line(x1+255,y1+255,x2+ 63,y2+ 63,c)
1140 endfunc
```



るボールを用意しているんですよ。リスト1でいうと1000行から1020行がボールの表示なんですが，ここで，

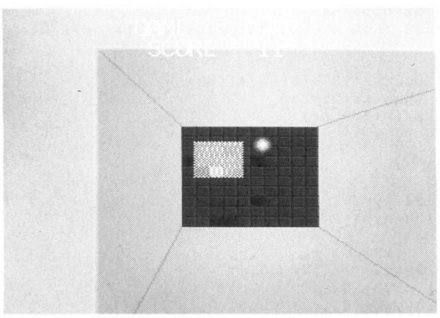
```
sp_set(4,x2,y2,bz/10+259,pr)
```

ってやってますよね。x2,y2がボールの表示される位置でbzがボールがどのくらい奥にいるかっていう変数なんです。

sp_set()は1番目の引数がスプライトプレーン，2，3番目がスプライト表示座標，4番目はパターンの番号で最後は優先度です。つまり，常にボールを表示するスプライトプレーンは4で，奥行きとパターン番号をうまく計算式でからめてボールの大きさを決めてるんですね。

ところで，もうひとつ不思議なことがあるんです。それはスプライトのPCGデータです。いっつもショートプログラムでは，スプライトPCGデータがネックになるんですよね。

だって1ドットが1文字でしょ。16×16ドットのキャラクター1個を定義するには16×16個で256文字。5，6個のキャラを作るのに1000個以上の文字を打ち込まなくちゃいけなくなります。これでは，あっという間にショートプロどころじゃない，デカプログラムになっちゃうんですよ。なんとかデータを減らしたいっていうとLHA. Xを使ってデータを縮める(これが結構縮みます)方法もあるんですけど，このプログラムはそんなデータがある様子が全然ありません。いったいどうやってるんだろう。


ちらつきを抑えるためには?

それさえわかればすごく楽かもっ! と考えてみましょう。スプライトのほかになにか絵を描くところっていうと……そう，グラフィックに絵が描けますよね。グラフィックならline()やcircle()，paint()関数を使うとsp_set()でデータを作るよりプログラムが小さくなっていいんですけど果たしてそんな命令があるかな? リスト1を探してみると，やっぱりあった。160～260行あたりにpaint()やcircle()やbox()が並んでました。ということはこのへんでグラフィックからスプライトPCGデータを作ってるに違いないですよね。

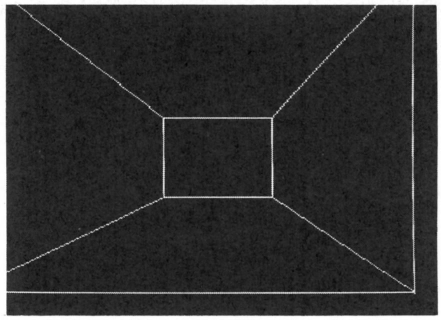
スプライトはsp_set()で配列に入ってるデータをスプライトPCGデータとして定義できますから，グラフィックを配列に取り込む方法があればいいはずです。

さっそく，リストの続きにスプライト関係の命令があるかどうか探していくと，見事，280行にありました。

```
280 get(i*16,0,i*16+15,15,buf):
sp_def(i,buf)
```

そう，これがスプライト定義を短くするタネだったんです。get()って関数をマニュアルをめくってみましょう。ふ～ん，なるほど。

普通はグラフィック画面からデータを取るっていうと，


大きさの違うパターンが定義されている

●point(x,y)

グラフィック画面上のドットのパレットコードを返します

を思い出してしまいますが，マニュアルのget()のページを見ると，

●get(x1,y1,x2,y2,na)

グラフィック画面の指定領域のドットパターンを，指定された配列naに読み込みます

という関数なんですね。

もしも，point()でsp_set()にデータを入れようとすると……pointだと1ドットが1つのデータですからforループで横に順に1つずつ拾っていきますね。それで，1つずつ配列に入れていけばいいから，以下のようになるはずです(データを入れる配列はchar dat(255)で定義されてます)。

```
for i=0 to 15
  for j=0 to 15
    dat(i*16+j)=point(j,i)
  next
next
sp_def(0,dat)
```

で，get()のほうは，マニュアルの下のほうにはこんなふうにも書いてあります。

●実画面サイズ512×512,256色モード

読み込み順(x1,y1)，(x1+1,y1)，……(x2,y1)，(x1,y1)，(x1+1,y1)，……(x2,y2)

char型配列の場合　＝1ドット単位
int型配列の場合　　＝4ドット単位
float型配列の場合　＝8ドット単位

そう，これって，256色でデータ型がint型の配列だったら1ドット単位でデータが順に入っていく，つまりsp_def()で使うデータの配列とまったく同じ形なんです。だから256色モードでget()を使うと……，

```
get(0,0,15,15,dat)
sp_def(0,dat)
```

これだけで，グラフィックに描かれたものをスプライトPCGとして取り込めちゃうんです。

おまけにこの方法だとプログラムが短いうえに，forループを使って何度も命令を実

行させる必要がないので，スピードが速く，大量のスプライト定義もすぐにできてしまいます。もう感動的ですらあるではないですか。うう。あ，ただしこの方法を使うには，グラフィックはパレット０〜15だけを使って描く必要がありますね，スプライトに使えるのは16色だけですから。

でも，これでスプライトは解決したといってもいいかな。

グラフィックのミソを引き出す

続いてはグラフィックのミソを引き出しましょう。このゲームではパドルもボールもそしてブロックも四角いチューブの中にいます。そして，この背景になるチューブはグラフィックで描かれていましたよね。

で，この背景ですが，ブロックのあるいちばん奥の部分は大きさが変わらないんですが，上下左右の壁の部分はパドル，つまり自分の視点の位置が変わるので微妙に書き換えなくてはいけないんです。ゲームスタートする前から背景は動きますので，ちょっとボタンを押す前にマウスを動かしながら，左側の壁，特に壁と上壁の境目の線に注目してください。ほら，パドルが壁に近づくと境目の線が急に，遠ざかると緩くなるのがわかるでしょ?

これを実際の物体で確認してみましょう。まず，なるべく大きく，先の長くて四角い箱を用意してみましょう。そして，中を覗いてみてください。先のほうの四角はどんなふうに見えますか? 手前の四角のなかに先の四角が小さく見えますよね。でもって，この箱を，目が左壁に近づくように，左のほうへずらしていくと……そう，手前と先の四角の大きさは変わらないけど，先の四角が左の壁に近づいてきますよね。

だから，こんな背景を描くには，

1) まず，手前の四角を描く
2) 次に先の四角を描く
3) 手前と先の四角の四隅の対応する線を描く

と，リスト２のようにやってやればいいはずですね。ではリスト２を打ち込んでRUNしてみましょう。

……はて? あれあれ? まだちょっと，具合がよくないですね。そう，なんだか画面がチラチラしちゃうんです。ゲームのほうでは絵がちらついていないのになぜこっちのリストはちらついてしまうんでしょうか。

実は，画面がちらついてしまうのは画面への絵の描き方に問題があるからなんです。リスト１のほうを見てみましょう。表２を見ると背景を描くのは720行〜750行までと1050行からのwall()関数でしたよね。wall()関数は，絵を描くのに関係ありそうなのはline()だけでリスト２といっしょみたいですよね。そうすると720行から順に見比べていくと，あれ，見なれない命令がありますね。

720 vpage(pa＊2＋11):pa＝(pa＋1) mod 2:apage(pa＋1)

そう，実はこのvpage()とapage()こそが絵をちらつかせないためのタネなんですね。リスト２の30〜50行を見てください。といっても30行のwhile 1と50行endwhileは「この間をずっとぐるぐるまわっていてね」という意味ですから実質的には40行だけに注目してください。

40 wall(x,y,0):mspos(x,y):wall(x,y,14)

wall()という関数は１番目と２番目の引数の位置に視点があるとき，３番目の引数の色で壁を描くっていう関数です。ということは，40行では，まずwall(x,y,0)でx,yの位置に０，つまり黒で壁を描くことになります。この壁以外の部分の色はもともと黒いですから……そう，黒で描くってことは「壁を消してる」ってことなんです。それからmspos(x,y)として，現在のマウス

図２ 視点がずれた場合の先の見え方

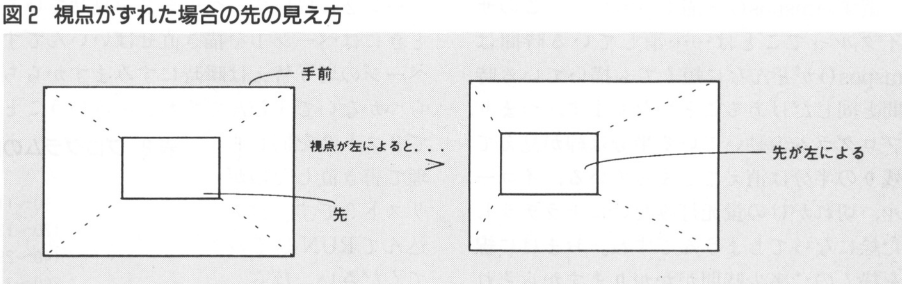
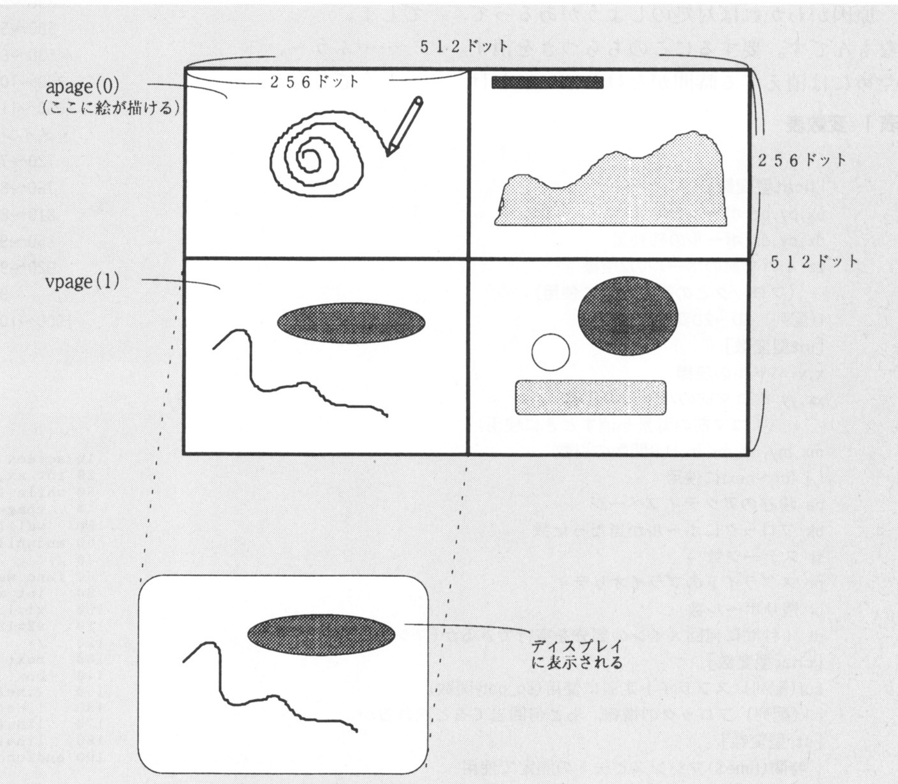


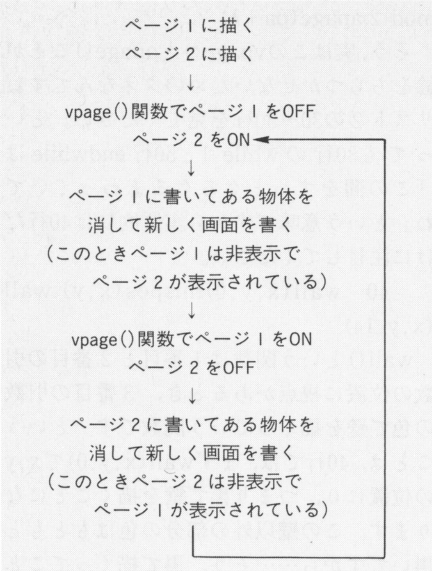
図３ apage()とvpage()



リスト2

```
 10 screen 0,1,1,1
 20 int x,y
 30 while 1
 40   wall(x,y,0):mspos(x,y):wall(x,y,14)
 50 endwhile
 60 /*
 70 func wall(wx,wy,c)
 80   int x1,y1,x2,y2
 90   x1=128-wx   :y1=128-wy
100   x2=128-wx/4:y2=128-wy/4
110   /*
120   box( x1    ,y1     ,x1+255,y1+255,c)
130   box( x2    ,y2     ,x2+ 63,y2+ 63,c)
140   line(x1    ,y1     ,x2    ,y2    ,c)
150   line(x1    ,y1+255,x2    ,y2+ 63,c)
160   line(x1+255,y1    ,x2+ 63,y2    ,c)
170   line(x1+255,y1+255,x2+ 63,y2+ 63,c)
180 endfunc
```

図4 ページ切り替えの様子



座標を得て，それからもう一度wall(x,y,14)で壁を描いているんです。

消す→mspos()→描く→消す……このサイクルってことは……消している時間はmspos()がどんなに短くても描いている時間と同じだけあることになります。つまり，プログラムの動いている半分は線が見えて残りの半分は消えてしまっている，イコール，切れかけの蛍光灯みたいにチラチラした絵になってしまうんですね。おまけに線を描くのに多少時間がかかりますからそれもちらつく原因になっています。

原因がわかれば対処のしようがあるってなもんです。要するにこのちらつきを消すためには消えてる時間がなければいいわけです。そこで登場するのが，先ほどのapage()とvpage()です。

X68000では絵を描くための場所であるG-RAMは256×256ドット，16色モードでは，グラフィック画面を4ページ分を確保できるのを知っていますか? つまり，4ページのうち「絵を描くページ」「絵を見せるページ」に分けることができるのです。それがapage()とvpage()という関数なんですね。詳しいことはマニュアルを読んでもらうとして，たとえばapage(1)ってすれば1ページ目を絵を描くページに，apage(2)で2ページ目に絵を描くようにできるんです。

もう理屈はわかっちゃいましたよね。そう，ちらつきを抑えるのに，背景を見せているページと描いているページを分けるのです。具体的には図4のような流れでプログラムを組むことになります。

つまり，ページ1を見せているときにはページ2に描いて，ページ2を見せているときにはページ1を描き直せばいいんです。ページの切り替えは瞬時にすみますからちらつかないですむんですね。そういうことでリスト2を同じ原理で書き直したのがリスト3です。打ち込んでRUNしてみてください。ほら，ちらつかなくなったでしょ。

……ふう～。

どうでしたか?

いろんなテクニックがわかったでしょう。ショートプログラム1本だけでもこれだけいろいろな技が詰まっているものなんです。だからショートプロで遊んでいるといろいろなテクニックが見えてきますよ，うんうん(宣伝モード)。

このプログラムにしても，まだまだほかにもいろいろテクニックが隠されていたりするんです。リストを打ち込み，リストを読んで，自分で見つけてくださいね。作者の高木さんが投稿原稿につけてくれた変数表（表1）とプログラムの簡単な説明（表2）もつけておきます(そうそう，今月の原稿の解説も実はかなり高木さんの投稿原稿によるところが大きかったりするのです。大感謝)。

さあ，皆さんもどんどんショートプロからいろいろなテクニックを盗んでみてくださいね! でもってついでに自分でプログラムを作ってみて，そいつをショートプロに投稿してくれるといいな，なんて思っています，はい。それではまた来月……じゃなくて，隣にある今月の「(で)のショートプロぱーてぃ」へいってみようか。

表1 変数表

[float型変数]
bx,by,bz ボールの座標
dx,dy,dz ボールの移動量
zz 1コマ前のボールのZ座標
(ブロックとの衝突判定に使用)
f(配列) 3D→2D変換に使用
[int型変数]
x,y パドルの座標
xx,yy 1コマ前のパドルの座標
(1コマ前の背景を消すときに使用)
mx,my,ml,mr msstat関数の引数
i,j for～nextに使用
pa 現在のアクティブページ
bk ブロックにボールが当たった数
st ステージ数
pr スプライトのプライオリティ
bl 残りボール数
dt 1秒間に何回メインの部分を実行できるか(マシンの速さ)
[char型変数]
buf(配列) スプライト定義に使用(sp_paty関数)
blk(配列) ブロックの情報，あと何回当てると壊れるか
[str型変数]
t 時間(time$)マシンスピードの測定で使用

表2 プログラムの簡単な説明

10～110	初期設定(256色モード)
120～130	効果音の定義
140～290	スプライトの作成，定義
300～350	カラーパレットの定義
360～380	初期設定(16色モード)
390～490	画面作成
500～520	マシンスピード測定
530～690	ゲームの始めから終わりまでのループ
700～1030	ゲームのメインルーチン
1050～1140	背景を描く
・メインルーチン内	
720～750	背景作成
760～800	ボールをまだ発射していないときの処理
810～840	ボールの移動，壁との衝突判定
850～910	ブロックとの衝突判定，処理
920～960	パドルとの衝突判定，処理
970	奥の壁との衝突判定
1000～1020	ボールの表示

リスト3

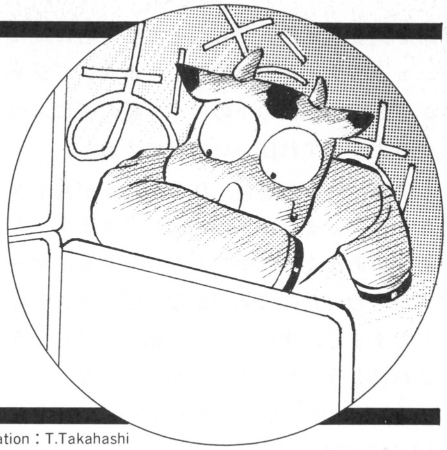
```
 10 screen 0,1,1,1
 20 int xx,yy,x,y,pa
 30 while 1
 40   vpage(pa*2+11):pa=(pa+1) mod 2:apage(pa+1)
 50   wall(xx,yy,0):xx=x:yy=y:mspos(x,y):wall(x,y,14)
 60 endwhile
 70 /*
 80 func wall(wx,wy,c)
 90   int x1,y1,x2,y2
100   x1=128-wx  :y1=128-wy
110   x2=128-wx/4:y2=128-wy/4
120   /*
130   box( x1    ,y1    ,x1+255,y1+255,c)
140   box( x2    ,y2    ,x2+ 63,y2+ 63,c)
150   line(x1    ,y1    ,x2    ,y2    ,c)
160   line(x1    ,y1+255,x2    ,y2+ 63,c)
170   line(x1+255,y1    ,x2+ 63,y2    ,c)
180   line(x1+255,y1+255,x2+ 63,y2+ 63,c)
190 endfunc
```

(で)のショートプロぱーてぃ――その53

今年最初のゲーム特集

Komura Satoshi 古村　聡

今月はタイトルにもあるとおり，３本ともゲームです。しかも全部がX-BASICで書かれています。なかには特別な外部関数を使っているプログラムもあるけれど，なんとか手に入れて，みなさん楽しんでください。



illustration：T.Takahashi

冷蔵庫，それは最後のフロンティア。

ひとりぐらしの冷蔵庫は怖いぞ～。なにせ下手すりゃひと月も開けないこともあるわけです。賞味期限がきれたオレンジジュースやカレーパンならまだしも，ちょっとした気の迷いで買ってしまった野菜サラダなんかは原形をとどめないくらい茶色のぐちゃぐちゃになっています。それはもう絵にも描けない恐ろしさだったりするのです。食事中の方がいたらごめんなさい。でも読みながら食べるなんてお行儀が悪いぞ。

さて，去年の年末のことなんですが，その冷蔵庫の大掃除をしていたら……見つけてしまったんですよ。製造年月日がちょうど１年前のジャワティーストレートのペットボトル。

見た感じ，変質している様子はないんですが……。それに「食べ物は粗末にしちゃいけませんっ！」と小さい頃から育てられているので捨てるのもためらわれるし……どうしたと思います？　しかたがないんで私，迷ったあげくに風呂を沸かしておもむろにジャワティーをどぼどぼーっと……ジャワティー風呂にしてしまったんですね。

いや～，大掃除で体中痛かったんですっごく気持ちよかったです。いや～，やっぱり食べ物は有効活用しなくちゃだよね。うんうん。

ボールボールで押し合え！

さて，今月はどういうわけだか，BASICのゲームっぽいものばかりが選ばれてしまったんですよ。厳密にいえば，ゲームではないものもあるけれど，今年最初のゲーム特集ってことでいいですよね。それでは，まず今月の１本目，ファイトでぶつかれ！DBBALL. BASです。どうぞ！

DBBALL.BAS for X68000

（X-BASIC　要ジョイスティック）

富山県　藤井栄一

このプログラムは対戦専用のぶつかりゲームです。

えーっと，このプログラムには準備するものがいろいろとあります。まずリスト１。このプログラムのリストですね。それとX68000。対戦する人が２人。それからジョイスティックが２本必要になります。

まず，X68000の電源を入れてBASICを立ち上げます。

A>BASIC

これでBASICが立ち上がりますね。

それからBASIC上からリスト１を打ち込んでください。間違いのないように入れていってくださいね。

入力が終わったら、いったんディスク上に打ち込んだプログラムを保存しましょう。

SAVE "DBBALL. BAS"

でディスクに書き込まれます。

さて，セーブが終わったら，

RUN

でプログラムを実行してください。

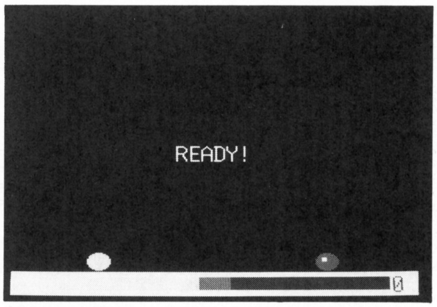
正しくリストが入力できていたら写真のような画面になるはずです。うまくいきましたか？　それでは遊び方を説明します。

写真の画面で，プレイヤー２人がジョイスティックをどこかの方向に倒すとゲームスタートです。キャラクターはボール。プレイヤー１は左の青い玉，プレイヤー２は右の赤い玉がマイキャラです。ジョイスティックを左右に動かすことで玉が動きだします。玉はジョイスティックを右に倒せば右に，左に倒せば左に加速度がつきます。玉は加速度がつくと慣性で動きつづけようとします。しかし，ある一定の速度まで達するとそれ以上速くなりません。

玉はジョイスティックのBボタンでジャンプします。上を押しながらだと高くジャンプできます。Aボタンを押すとその時点での速度が２倍になります。その代わり少しLIFEを消費します。対戦で相手のLIFEポイントを完全になくすと勝ちです。はやい話「はっけよいのこった」みたいなものなんですね，はい。

んー，も，も，燃える～。直観的には自分が相手をやっつけたのか，相手にやられたのかわかりにくいけど，それだけに相手をやっつけようと右へ左へジョイスティックをガッツンガッツン倒してしまいますね。あんまり熱中してるもんだから，玉の動きといっしょに体までがっつんがっつん動いちゃうわ，対戦相手の肩とぶつかっちゃうわ……女の子とやったりしたら肩と肩で……ああっ，いいかもしれない(なにひとりでバカいってんだか)。なにはともあれ激しいゲームです。

あ，相手の倒し方なんですが，なるべく速く相手にぶち当たってください。垂直方向水平方向，それぞれの加速度成分の絶対値を足したものの合計が少なかったほうがダメージを受けます。さらにその差が大きいほど大きなダメージを受けます。だから



DBBALL. BAS

敵を倒そうと思ったら，とにかく右に左にジョイスティックを動かしてボタンでジャンプして，画面の下のほうで敵の玉にぶつかるようにすればいいわけですね。

決着がついた場合，負けたほうがAボタンを押すとまた写真の画面になります。あぁー，あんまり熱中するもんだから，ゲームが終わったのに気づかないでボタンを押したままで，次のゲームが始まってしまう。これはどうにかしてほしかったかな。

ま，なにはともあれ燃えるゲームDBBALLなのでありました。

今度のさまよいは広い！

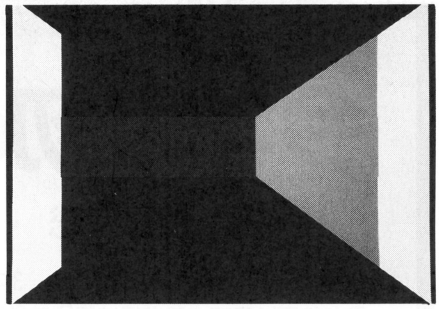
ではでは，まだまだアツいうちに次のプログラムにいってしまいましょう。今月2本目のプログラムはさまよって3D,WALK_3D.BASです。どうぞ！

WALK_3D.BAS & IMAZE.BAS
for X68000　　(X-BASIC，要APIC.FNC)

岡山県　杉浦竜夫

あ，これはゲームではなくて，3D表示の迷路なんですね。縦（510）×横（510）のでっかい迷路をさまよいます。

このプログラムを実行するには迷路を表示するための迷路データが必要になります。まずはX-BASICを立ち上げてからリスト2のIMAZE.BASを入力してください。これは，3D迷路のためのデータを生成するためのプログラムです。迷路のデータは，MAZE. PICというファイル名でディスクに作られます。その際にこのプログラムではAPIC. FNCを使っています。APIC.FNC創刊は「10周年記念PRO-68K」（1992年6月号）のディスクの中に入っています。



IMAZE. BAS & WALK_3D. BAS

APIC. FNCをBASICのディレクトリにコピーしてから，BASICの設定ファイル（BASIC.CNF）に，

　FUNC＝APIC

と書き加えて，X-BASICからPIC関係の命令が使えるようにしておいてください（持っていない人はIMG関係の命令に換える。ファイル名は“mage.gm3”にすること）。

このプログラムは迷路を生成するのに40

リスト1　DBBALL. BAS

```
 10 /*/*/*/*/*/*/*/*ｻｻﾉﾐﾁﾙﾌｯｶﾂｷﾈﾝ?/*/*/*/*/*/*/*/*/*/*/*/*/*
 20 /*    DBBALL. BAS     1993/11/3
 30 /*
 40 /*                              programmed by 藤井栄一
 50 /*
 60 /*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*
 70 screen 0,2,1,1:console ,,0
 80 dim char g(255):dim int life(1)={100,100},win(1)
 90 dim str n(1)={"RED","BLUE"}
100 int x1=48,y1=220,j1,inter1,dx1,dy1
110 int x2=192,y2=220,j2,inter2,dx2,dy2
120 int a,b,c,d,dinter,death
130 sp_init():sp_disp(1):sp_on()
140 for i=0 to 1
150 circle(7,7,7,1+i):fill(4,4,6,6,255):paint(1,7,1+i)
160 get(0,0,15,15,g):sp_def(i+1,g,1):sp_color(1+i,62+i*1922)
170 next:wipe():mus()
180 while 1
190     gamen()
200 repeat
210     t=stick(1):s=strig(1):inter1=inter1-1
220     if abs(dx1)>12 then dx1=dx1-3*(dx1/abs(dx1))
230     if s=2 and inter1<=0 then{
240         dx1=dx1*2:dy1=dy1*2:inter1=20
250         life(0)=life(0)-4:fi(1)}
260     if x1+dx1<0 or x1+dx1>240 then {dx1=-dx1
270                     }else dx1=dx1+2*((t=4)-(t=6))
280     if s=1 and j1=0 then dy1=-20+6*(t-6>0):j1=1
290     if j1=1 then dy1=dy1+2:y1=y1+dy1:m_play(1)
300     if y1+dy1<0 then dy1=-dy1
310     if y1>220 then y1=220:j1=0:m_play(2)
320     x1=x1+dx1:sp_move(1,x1,y1,1)
330 /*
340     t=stick(2):s=strig(2):inter2=inter2-1
350     if abs(dx2)>12 then dx2=dx2-3*(dx2/abs(dx2))
360     if s=2 and inter2<=0 then{
370         dx2=dx2*2:dy2=dy2*2:inter2=20
380         life(1)=life(1)-4:fi(2)}
390     if x2+dx2<0 or x2+dx2>240 then {dx2=-dx2
400         }else dx2=dx2+2*((t=4)-(t=6))
410     if s=1 and j2=0 then dy2=-20+6*(t-6>0):j2=1
420     if j2=1 then dy2=dy2+2:y2=y2+dy2:m_play(1)
430     if y2+dy2<0 then dy2=-dy2
440     if y2>220 then y2=220:j2=0:m_play(2)
450     x2=x2+dx2:sp_move(2,x2,y2,2)
460 /*
470     a=x1-x2:b=y1-y2:dinter=dinter-1
480     if abs(a)<15 and abs(b)<15 and dinter<0 then {
490         m_play(3)
500         c=((abs(dx1)+abs(dy1))-(abs(dx2)+abs(dy2)))
510         dx1=-dx1:dy1=-dy1:dx2=-dx2:dy2=-dy2
520         if c=0 then continue
530         life(-(c>0))=life(-(c>0))-abs(c):dinter=10
540         palet(0,rgb(-31*(c>0),0,-31*(c<0)))
550         fi(1-(c>0)):palet(0,0)
560         if life(-(c>0))<0 then death=1-(c>0)
570     }
580 /*
590 until death>0
600 win(-(death=1))=win(-(death=1))+1
610 locate 12,0:print n(-(death=2))+" WIN"
620 repeat:until strig(1-(death=2))=1
630 x1=48:y1=220:x2=192:y2=220:dx1=0:dx2=0:dy1=0:dy2=0
640 life(0)=100:life(1)=100:inter1=0:inter2=0:dinter=0
650 death=0
660 endwhile
670 /*
680 func mus()
690 for i=1 to 3:m_alloc(i,100):m_assign(i,i):next
700 m_trk(1,"a@38"):m_trk(2,"a@48"):m_trk(3,"a@62")
710 endfunc
720 func gamen()
730 sp_move(1,x1,y1,1):sp_move(2,x2,y2,2)
740 cls:palet(255,rgb(25,25,25)):fill(0,236,255,256,255)
750 fill(18,240,117,250,7):fill(138,240,237,250,40)
760 fill(118,240,137,250,20)
770 symbol(1,239,itoa(win(0)),1,1,1,7,0)
780 symbol(239,239,itoa(win(1)),1,1,1,40,0)
790 locate 13,8:print "READY!"
800 repeat:until stick(1)<>0 and stick(2)<>0:cls
810 endfunc
820 func fi(a;int)
830 if a=1 then {fill(18,240,118-life(0),250,0)
840     }else{fill(238,240,138+life(1),250,0)}
850 endfunc
```

リスト2

```
 10 int x=2,y=2,tx=4,ty=2,rn,o,i(2)
 20 screen 1,2,1,1:console,,0
 30 box(0,0,510,510,1):palet(1,65535)
 40 while 1
 50    o=0
 60    if point(x,  y-2)=0 then i(o)=0:o=o+1
 70    if point(x+2,y  )=0 then i(o)=1:o=o+1
 80    if point(x,  y+2)=0 then i(o)=2:o=o+1
 90    if point(x-2,y  )=0 then i(o)=3:o=o+1
100    if o=0 then {
110                for ty=ty to 508
120                for tx=tx to 508
130                    if point(tx,ty)=0 then pset(tx,ty,1):x=tx:
y=ty:tx=tx+2:o=4:break
140                tx=tx+1:next:if o=4 then break else tx=2
150                ty=ty+1:next:if o=4 then continue
160                apic_save("maze.pic",0,0,511,511):end
170               }
180 rn=rnd()*o
190  switch i(rn)
200    case 0:box(x  ,y-1,x,  y-2,1):y=y-2:break
210    case 1:box(x+1,y,  x+2,y,  1):x=x+2:break
220    case 2:box(x  ,y+1,x,  y+2,1):y=y+2:break
230    case 3:box(x-1,y,  x-2,y,  1):x=x-2:break
240  endswitch
250 endwhile
260 end
```

▶岡村さん，姉弟4コマ終わっちゃうんですか？　しくしく。　須藤　透(22)京都府

分ほどかかりますので，Cコンパイラを持っている人は，

A>CC IMAZE. BAS

でコンパイルしてIMAZE. Xを作ってから，

A>IMAZE

で迷路データを作ったほうがいいでしょう。ちなみにコンパイルすると10MHzのX68000なら1分ほどで作ることができるはずです。

さて，迷路データが完成したら今度は迷路をさまよいましょう。BASICを立ち上げた状態でリスト3，WALK_3D. BASを入力してRUNです。カーソルキーで移動できます。

へ～，3D迷路っていうとこれまでもぱーてぃハンズでとりあげたりしました。そのときは，表示がワイヤーフレームでデータはプログラム中に配列の形で置いていたんです。このプログラムでは迷路の表示はパレットを使ってサーフェス(表面)表示，データはプログラムで生成して，グラフィック画面に格納しています。操作性はいいし，表示は本格的，データはスマートに記録できるし，簡単に書き換えがきくし，頭のいい方法で作ったもんですね～。コンパイルしちゃうとRED ZONEだと速すぎちゃうくらいです。リストも短いし，感心感心。

ただ，この迷路には出口がないので注意してください。それを知らないと一生迷路でさまようことになりますからね。

さあ，どんどん打ち込んで3Dしてしまいましょう。

穴の開いた壁なのだっ

さてさて，いよいよ今月最後のプログラム。3本目のプログラムはクォータービュー3Dアクションゲーム，三重県の平井さんの作品でVOID. BASです。どうぞ！

VOID.BAS for X68000
(X-BASIC，要ジョイスティック，MUSICZ. FNC)
三重県　平井栄治

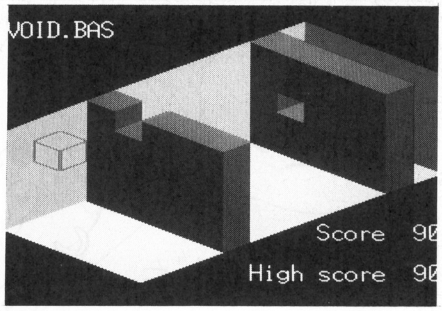
今月は3本ともX-BASIC用のリストですが，このプログラムはちょっと下準備がいるんです。このプログラムではZ-MUSICが必須ですのでコマンドラインから，

A>ZMUSIC

としてZ-MUSIC本体を常駐させます。ほかにも，BASICにはZ-MUSIC用の外部関数ファイルMUSICZ. FNCを設定する必要があります。設定方法ですが，MUSICZ. FNCをBASICのディレクトリにコピーします。それからBASICの設定ファイル，BASIC. CNFに，

FUNC＝MUSICZ

と書き込んでBASICを立ち上げればOKです。MUSICZ. FNCはZ-MUSICシステムのなかにあります。あ，ゲームをプレイするにはジョイスティックも忘れないでくださいね。



VOID. BAS

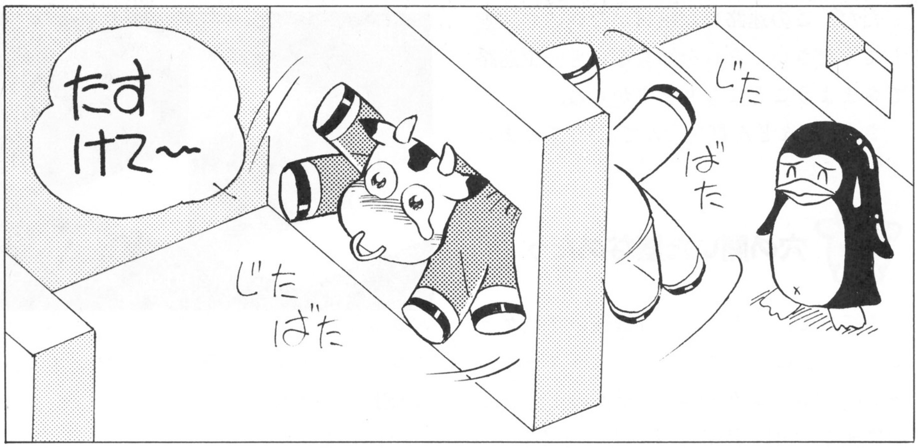
さてさて，リスト4を入力して遊びましょう。RUNすると，ちょっと間があってから，青い通路と緑のワイヤーフレームの自機(って単なる立方体なんですけどね)を斜め上から見た状態，つまりクォータービューのゲーム画面が表示されます。ジョイスティックのボタンを押すとゲームスタート。

赤い壁が進路方向からスクロールしてきます。壁には1カ所だけ穴が開いています

リスト3　WALK_3D. BAS

```
 10 int px(9)={-1, 1,-1, 0, 1,-2,-1, 0, 1, 2}
 20 int py(9)={ 0, 0,-1,-1,-1,-2,-2,-2,-2,-2}
 30 int  s(3)={ 0, 1, 0,-1}
 40 int  c(3)={ 1, 0,-1, 0}
 50 int col(17)={0,28734,28734,10262,28734,28734,
 60             +0,10262,18474,18474,28734,28734,28734,28734}
 70 int x=1,y=1,dr=1,p(9)
 80 screen 1,2,1,1:console,,0
 90 line(  0,  0, 60 ,60, 2)
100 line(  0,511, 60,450, 2)
110 line(450,450,511,511, 3)
120 line(511,  0,450, 60, 3)
130 line( 60,450,450,450, 4)
140 line( 60, 60,450, 60, 4)
150 line( 60, 60,200,200, 9)
160 line( 60,450,200,300, 9)
170 line(300,300,450,450,10)
180 line(450, 60,300,200,10)
190 fill(200,200,300,300, 1)
200 fill(  0, 60 ,60,200, 5)
210 fill(  0,300, 60,450, 5)
220 fill(450,300,511,450, 6)
230 fill(450, 60,511,200, 6)
240 fill( 60,200,200,300,11)
250 fill(300,200,450,300,12)
260 paint( 10, 20, 2)
270 paint( 10,500, 2)
280 paint(510, 22, 3)
290 paint(508,490, 3)
300 paint(250, 70, 4)
310 paint(250,430, 4)
320 paint( 10,255, 7)
330 paint(508,255, 8)
340 paint( 70, 80, 9)
350 paint( 70,400, 9)
360 paint(410,160,10)
370 paint(410,330,10)
380 apage(1):apic_load("maze.pic",0,0)
390 vpage(1):drow()
400 while 1
410         switch asc(inkey$(0))
420                 case 30:forth()         :break
430                 case 31:back()          :break
440                 case 28:t_l()           :break
450                 case 29:t_r()           :break
460                 case 27:end
470         endswitch
480 endwhile
490 end
500 func drow()
510         for i=0 to 9        /*次の座標の壁の有無を配列に入れる。
520                 p(i)=point(x+px(i)*c(dr)-py(i)*s(dr),y+px(i)
*s(dr)+py(i)*c(dr))
530         next                /*パレットを切り替える。
540         palet( 1,col(   p(3)+p(7)*3               ))
550         palet( 2,col(   p(0)                      ))
560         palet( 3,col(   p(1)                      ))
570         palet( 4,col(   p(3)                      ))
580         palet( 5,col(   p(0)+p(2)                 ))
590         palet( 6,col(   p(1)+p(4)                 ))
600         palet( 7,col(   p(0)+p(2)+p(5)*3          ))
610         palet( 8,col(   p(1)+p(4)+p(9)*3          ))
620         palet( 9,col(   p(2)*2+p(3)*4         +6 ))
630         palet(10,col(   p(4)*2+p(3)*4         +6 ))
640         palet(11,col(   p(6)+p(2)*2+p(3)*4    +6 ))
650         palet(12,col(   p(8)+p(4)*2+p(3)*4    +6 ))
660 endfunc
670 func forth()
680         if point(x+s(dr),y-c(dr))=1 then {
690                 locate 30,15:print "いたい!"
700                 locate 30,15:print "       "
710                 return()
720         }
730         x=x+s(dr)
740         y=y-c(dr)
750         drow()
760 endfunc
770 func back()
780         if point(x-s(dr),y+c(dr))=1 then {
790                 locate 30,15:print "いたい!"
800                 locate 30,15:print "       "
810                 return()
820         }
830         x=x-s(dr)
840         y=y+c(dr)
850         drow()
860 endfunc
870 func t_l():dr=dr+1:if dr=4  then {dr=0} :drow():endfunc
880 func t_r():dr=dr-1:if dr=-1 then {dr=3} :drow():endfunc
```

▶ジャストはなにをする気だろう。なんかすごいぞ。満開製作所みたいに怪しくていいぞ。
尾崎　浩(28)島根県



ので，ジョイスティックの上下左右で自機を操作してぶつからないように通していってください。100個の壁を通過するか，壁に激突するとゲームオーバーです。

うおお！　む，難しい〜！

私，昔あったクォータービューのゲーム，「ザクソン」とかちょっと得意だったはずだったんですけど，こ，これは難しいですわ。っていうのも，このゲームは高さがあるせいで穴が真ん中あたりにあると穴より斜め上にいても穴ぴったりの位置にいるような錯覚を覚えてぶつかったりしてしまうんですよ。それとちょっと操作感が重いかなって感じがあるんで(16MHzあたりだとかなり楽にはなるけど)ゲームとしてはかなりスリリングです。ああ，今月は私の愛用スティックXE-1proがボロボロになっちゃうよー。もう売ってないのに。

これで自機の影でも地面に映っていたらずいぶん操作が楽なのにー，と思ってしまいます。今度作るときは，その辺も考えてみてね。

さぁてっと，風呂あがりにガンガンゲームをしたらなんだかノドが乾いてしまったなー。冷蔵庫のなかになんか入ってないかなあ……ああっ，片付け忘れた野菜カゴにカルピスの原液1リットルビンが！　しかも賞味期限は1年前だ！　こ，ここれは……カ，カル……ピス風呂？

ううっ，また来月。

リスト4　VOID. BAS

```
10 screen 0,1,1,1:sp_init():sp_disp(1):dim       z(225)     ={
20                         0,192,240,252,255, 63, 15,  3, 12,
30                        48,128,176,140,131,193, 49, 13,  1,
40          0, 80, 16, 72, 32, 64, 48, 56, 64, 48, 80, 40,
50         96, 32,112, 24,128, 16,144,  8,160,  0,256,  0,
60        256,336,272,328,288,320,304,312,320,304,336,296,
70        352,288,368,280,384,272,400,264,416,256,256,  0,
80          4, 8, 1, 1, 5, 9, 1, 1, 6, 3, 8, 1, 7, 3, 9, 1,
90          3, 8, 1, 1, 3, 9, 1, 1, 6,10, 2, 2, 7,11, 2, 2,
100         1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
110         2, 2, 2, 2, 2, 2, 2, 2, 1, 1, 1,12, 1, 1, 1,13,
120         1, 1, 1,12, 1, 1, 1,13, 2, 2, 2,14, 2, 2, 2,15,
130         0, 0, 4, 8, 0, 0, 5, 9, 0, 4, 8, 1, 0, 5, 9, 1,
140        10, 2,18, 8,11, 2,19, 9, 0, 0, 6,10, 0, 0, 7,11,
150         1,12, 4, 8, 1,13, 5, 9, 1,16,18, 8, 1,17,19, 9,
160         2,14, 6,10, 2,15, 7,11, 1,12, 0, 0, 1,13, 0, 0,
170         1,16,14, 0, 1,17,15, 0, 2,14, 0, 0, 2,15, 0, 0}
180   g=0:h=0:i=0:j=0:k=0:m=0:n=0:o=0:p=0:q=0:r=0:s=0:t=0:u=0
190 m_init():m_tempo(181):m_alloc(1,81):dim char y(255)
200 m_assign(1,1):v=0:w=0:m_trk(1,"r8"):dim  str x( 11)[32]={
210              "00001234","12344444","76544444","00007654",
220              "43210000","44444321","44444567","45670000",
230                         "abajaiahbajaiahahaqapaoarhrirjrb",
240                         "haiajabaahaiajababalamanbkjkikhk",
250                         "rkrkrkrkrkrkrkrkhkikjkbkanamalab",
260                         "rkrkrkrkrkrkrkrkrbrjrirhoapaqaha"}
270              a(0,0,0):a(0,1,1):a(0,2,2):a(0,3,3):a(0,0,3)
280              a(1,0,3):a(2,0,3):a(3,0,3):a(4,1,3):a(5,1,3)
290              a(6,2,3):a(7,2,3):a(4,0,1):a(5,0,1):a(6,0,2)
300              a(7,0,2):a(4,2,1):a(5,2,1):a(0,2,3):a(1,2,3)
310                 for v=8 to 11:    u=0       :for t=1 to 32
320                 s=z(asc(mid$(x(v),t,1))-97):for r=1 to  8
330              q=s/2:q=q+q: if s=q then{y(u)=0}else{y(u)=4}
340              s=s/2:u=u+1:next:next:sp_def(4*v+w,y,1):next
350         sp_on():for w=0 to  1: bg_fill(w,0):bg_set(w,w,1)
360       bg_scroll(w,0,256):next:for w=0 to  1:for v=0 to 32
370   for u=0 to  4:for t=0 to  4:       d( 66,w, v,   u,   t)
380        next:next:v=v+31:next:next:w=255:sp_color(4,49152)
390     sp_color(1, 1024):sp_color(2, 1536):sp_color(3, 1984)
400        palet(1,   32):   palet(2,   48):   palet(3,   62)
410     g= 16:i= 56:k= 80:n=103:p=135:r=175:t=183:u=184:v=223
420     h= 55:j= 79:m= 96:o=104:q=136:s=176:  line(k,v,w,q,3)
430       line(s,m,s,g,1):  line(s,g,w,h,1):  line(w,i,w,p,1)
440       line(0,t,0,o,2):  line(0,n,r,g,2):  line(r,g,r,m,2)
450       line(r,m,0,t,3):  line(0,u,j,v,3):  line(w,p,s,m,3)
460 g=3: paint(     v,o,1): paint(     8,o,2): paint(     8,u,3)
470 z(1)= 0:z(4)= 0:z(7)= 0:    locate  0, 1:print"VOID.BAS   "
480 z(2)= 0:z(5)= 0:z(8)=23:    locate 18,11:print"       Score"
490 z(3)= 0:z(6)= 0:z(9)=23:    locate 18,13:print"High score"
500 j=0:k=2:m=2:n=0:o=0:p=0:                      a(8,0,0): b()
510 while          p<8181:p=0:   locate  0, 3:print"Push.":beep
520 while strig(1)=    0:f():if p>0 then{break}else{}:endwhile
530 while          p=   0:       locate  0, 3:print"     ":beep
540     j=0:n=0:a(8,0,0):for i=0 to 1:bg_scroll(i,0,256):next
550              d( 66,0, 0,z(0),z(1)):d( 66,0,32,z(2),z(3))
560              d( 66,1, 0,z(4),z(5)):d( 66,1,32,z(6),z(7))
570 z(0)=f():z(1)=f():z(2)=  0:z(3)=  0:d(146,0, 0,z(0),z(1))
580 z(4)=f():z(5)=f():z(6)=  0:z(7)=  0:d(146,1, 0,z(4),z(5))
590 z(8)=  0:      for j=1 to 6:z(9)= 23:b():if p>0 then{break
600             }else{}:next:while p=0:b():endwhile:endwhile
610                     endwhile:beep:print chr$(11):end
620 func a(v,u,t): if      v=    8 then{         if j>  6 then{
630                 if 5*m+k=5*t+u then{n=n+10:if n>999 then{
640 n=999:p=1818}else{}:if n>o then{o=n}else{}}else{}}else{}
650 locate 29,11:print right$("   "+itoa(n),3)
660 locate 29,13:print right$("   "+itoa(o),3)+chr$(11)}else{
670  s=0:for r=1 to 8:q=z(atoi(mid$(x(v),r,1))):for p=1 to 8
680             o=q/2:o=o+o: if q=o then{y(s)=u}else{y(s)=t}
690        s=s+1:q=q/2:next:next:sp_def(w,y,0):w=w+1}:endfunc
700 func b():for i=1 to g:m_play(1):h=stick(1):switch      h
710 case 1:case 4:case 7:k=k-1:if k=-1 then{k=0}else{}:break
720 case 3:case 6:case 9:k=k+1:if k= 5 then{k=4}else{}:break
730 endswitch:                                  switch      h
740 case 7:case 8:case 9:m=m-1:if m=-1 then{m=0}else{}:break
750 case 1:case 2:case 3:m=m+1:if m= 5 then{m=4}else{}:break
760 endswitch:     sp_set(0,16+16*k,112+16*m+k*8,308,3)
770                sp_set(1,32+16*k,112+16*m+k*8,312,3)
780                sp_set(2,16+16*k,128+16*m+k*8,316,3)
790                sp_set(3,32+16*k,128+16*m+k*8,320,3)
800                    if i=g then{for h=0 to 1:switch z(8+h)
810               case 18:d( 66,h, 0,z(0+4*h),z(1+4*h)):break
820               case  6:d( 66,h,32,z(2+4*h),z(3+4*h)):break
830               case 21:              z(0+4*h)=       f()
840 z(1+4*h)=f()        :d(146,h, 0,z(0+4*h),z(1+4*h)):break
850               case  9:              z(2+4*h)=       f()
860 z(3+4*h)=f()        :d(146,h,32,z(2+4*h),z(3+4*h)):break
870 endswitch:          next}else{}:for h=0 to 1:switch z(8+h)
880               case 11:c(            z(0+4*h),z(1+4*h)):break
890               case 23:c(            z(2+4*h),z(3+4*h)):break
900 endswitch:          next:                  while m_stat()<>0
910 f():if p>0 then{break}else{}:                       endwhile
920 f():if p>0 then{break}else{}:next:if p>0 then{return(18)
930   }else{}:  bg_scroll(0,z(18+2*z(8)),z(19+2*z(8)))
940             bg_scroll(1,z(18+2*z(9)),z(19+2*z(9)))
950                                for h=0 to 1:switch z(8+h)
960               case 10:a(       8,z(0+4*h),z(1+4*h)):break
970               case 22:a(       8,z(2+4*h),z(3+4*h)):break
980 endswitch:next:z(8)=z(8)+1:if z(8)=24 then{z(8)=0}else{}
990                z(9)=z(9)+1:if z(9)=24 then{z(9)=0}else{}
1000                                                  endfunc
1010 func c(w,v):     if j>6 then{if 5*m+k=5*v+w then{
1020                             }else{p=1818}}else{}:endfunc
1030 func d(w,v,u,t,s):switch 5*s+t:case  0: e(w,v,u,t,s,0,0)
1040      e(w,v,u,t,s,2,1):   break: case  1: case  2: case  3
1050      e(w,v,u,t,s,0,2): e(w,v,u,t,s,2,1):   break: case  4
1060      e(w,v,u,t,s,0,2): e(w,v,u,t,s,2,3):   break: case  5
1070      case 10: case 15: e(w,v,u,t,s,0,4):   break: case 20
1080      e(w,v,u,t,s,0,7):   break: case 24: e(w,v,u,t,s,2,9)
1090               case 21: case 22: case 23: e(w,v,u,t,s,0,8)
1100         break:case  9: case 14: case 19: e(w,v,u,t,s,2,6)
1110               default: e(w,v,u,t,s,0,5):endswitch:endfunc
1120 func e(w,v,u,t,s,r,q)
1130   bg_put(v,20+u+t+t+r,12+u+t+s+s,256+z(0+8*q+w))
1140   bg_put(v,20+u+t+t+r,13+u+t+s+s,256+z(1+8*q+w))
1150   bg_put(v,20+u+t+t+r,14+u+t+s+s,256+z(2+8*q+w))
1160   bg_put(v,20+u+t+t+r,15+u+t+s+s,256+z(3+8*q+w))
1170   bg_put(v,21+u+t+t+r,12+u+t+s+s,256+z(4+8*q+w))
1180   bg_put(v,21+u+t+t+r,13+u+t+s+s,256+z(5+8*q+w))
1190   bg_put(v,21+u+t+t+r,14+u+t+s+s,256+z(6+8*q+w))
1200   bg_put(v,21+u+t+t+r,15+u+t+s+s,256+z(7+8*q+w)):endfunc
1210 func f():if inkey$(0)=chr$(27) then{p=8181}else{
1220                                }:return(5*rnd()):endfunc
1230 /* VOID              BAS        7751  93-10-27  12:00:00
```

▶1月号の「THE SENTINEL」のように，まだS-OSを使っている人に役立つような記事が毎月載るとうれしい。　前田　育男(19)東京都

ぷろぐらむ風まかせ 2

来月は他の入力装置……とかいっておきながら，実は空中３回転ひねりを加えて出力にいってしまいたいんだけどなー。だってさー，やっぱり絵を描いたり字を書いたりするほうが目に見えるから面白いじゃぁないですか。

ま，なにはともあれ今月はマウス以外の入力装置についての解説なのであります。ちょっと退屈かもしれないけど，がんばってついてきてね。

キーボードを使う

今月は，いろいろ種類はあるけれどいつも使っているものと違うと「()の位置が違うぞ〜!」「だー，親○シ○ト使えねーっ!」などとアビキョウカンの世界をかもしだすもの。パソコンを持っていれば誰でも持っていて，誰でも使える(……はず)のものが，このキーボードですよね。今月はまず，このキーボードを使ってみるのであります。

で，このキーボードを使うにあたって，それ関係の命令っていうのをX-BASICのマニュアルで調べてみるんですが……結構いろいろありますよね。INPUT，INKEY$……。ゲームなどに使うときにはどの命令を使うのがいいんでしょう?

ゲームで使う，ということは普通は「どのキーが，現在押されているか調べる」，つまりリアルタイムキー入力っていうのを使いたいわけですけど，他の機種ではINKEY$を使うことが多いのですが，X68000の場合，マニュアルを見てみるとINPUTもINKEY$も「入力があるまで待つ」命令みたいなんですね。なんだかこれでは使えないですね。

実は，1991年７月号の特別付録でついてきたX-BASICポケットリファレンスなどには書いてあるんですが，X-BASICではなぜか，リアルタイムキー入力は隠し命令になっているのです。

X-BASICではINKEY$(0)という命令で，いまキーボードでなにが押されているかがわかります。文字列型の変数をA$として用意してやって，

A$＝INKEY$(0)

とすると，A$にキーボードで押された内容が返ってきます。ちなみにA$＝INKEY$だとキーボードから文字が入力されるまで待っています。

だから，

WHILE(""＝INKEY$(0))：ENDWHILE

なんてするくらいなら，

A$＝INKEY$

としましょうね。

で，無事解決したかのように見えるリアルタイムキー入力なんですが，実は，このINKEY$(0)でキーボードを見る方法にはひとつ欠点があるんです。それは「いくつかのキーをいっぺんに押されると先に押されたキーがわからない!」ということなのです。たとえば，シューティングゲームを作っているときに，４と６のキーで自機の移動，スペースで弾を撃つとします。この場合に４とスペースのキーをいっぺんに押しても，コンピュータは一瞬遅く押されたほうのキーだけを「押されている」と判断してしまうんです。だから，弾を撃つと自機の動きが止まってしまうのです。

さーて，どうしたもんでしょう?

ジョイスティック

こーんなときにお役にたつのがジョイスティックだったりします。入力装置最後の解説はこのジョイスティックです。

実はジョイスティックはX-BASICを使うときには，とっても扱いやすいデバイス(あ，入力とか出力，それから記録の装置のことをデバイスっていうんですね)だったりするのです。キーボードと違ってスティックとボタンがいっぺんに押されても，もちろんボタンが２ついっぺんに押されてもわかりますし，X-BASICの命令も簡単なのです。

X-BASICでジョイスティックの操作は，STICK()，STRIG()命令を覚えておけばOKです。

STICK()命令は，ジョイスティックがどの方向に倒れているかを知ることができる命令です。

A＝STICK(1)

とするとジョイスティック１(PRO以外のマシンならパソコン本体の前側についてるコネクタにつないだジョイスティック)どの方向に倒れているかがAに入ります。倒れていなかったら０。それ以外のときにはそれぞれ倒れた方向について，

７８９
４＊６
１２３

と，上記のような値が返ってきます。()の中に２と書くとジョイスティック２の状態です。

もうひとつのSTRIG命令も簡単で，

A＝STRIG(1)

とすると，ジョイスティック１のボタンの押された状態がAという変数に入ります。状態っていうのは，ボタンが押されていないときには０。ボタンAが押されたときは１。ボタンBが押されたときは２。両方のボタンが押されたときは３という数字が返ってきます。

基本的にはジョイスティックというのは，ボタンとスティックを同時に押しても平気っていう以外は，ほとんどINKEY$(0)を使ったときのキーボードと扱いがいっしょなんですね。左か右か上か下かで，返ってくる数字もテンキーを押したときのINKEY$(0)に返ってくる値と同じですしね。

で，この長所ばかりのようなジョイスティックなんですが，しいて欠点を挙げるとすると，X68000では標準でジョイスティックがついてくるわけではないので，誰もが持っているとは限らないっていうことぐらいでしょうかね(ほとんどみんな持ってるような気もするけど……)。

サンプル

……ということで，マウス，キーボード，ジョイスティックを使ったサンプルです。BASICからリストを入力してRUNしてみてください。それからキーボード，ジョイスティック，マウスをいろいろ触ってみてください。それぞれにいろいろなデータが入力されているのがわかるでしょう。あまり派手な表示じゃないですけど，自分の目で確認してください。

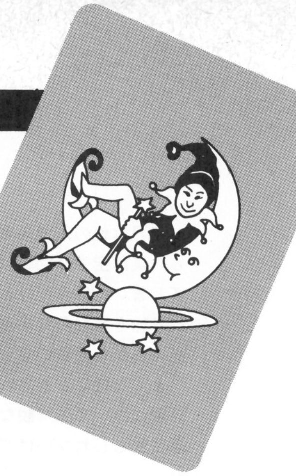
さ，来月からは出力に入ります。画面出力にしようかな〜。では，また来月。

```
 10 str a$
 20 int st,trig
 30 int dummy,msx,msy,msbl,msbr
 40 width 64:mouse(4):mouse(1)
 50 while(1)
 60   a$=inkey$(0)
 70   msstat(dummy,dummy,msbl,msbr)
 80   mspos(msx,msy)
 90   st = stick(1)
100   trig=strig(1)
110   if a$<>"" then{
120     locate 0,0:print"キーボード ["; a$;"]"
130   }else{
140     locate 0,0:print"キーボード [ ]"
150   }
160   locate 0,1:print"マウス X=";msx;" Y=";msy
170   locate 0,2:print"        ボタン右=";msbr;" 左=";msbl;
180  locate 0,3:print"ジョイスティック 方向";st
190  locate 0,4:print"                 ボタン=";trig
200 endwhile
```

▶犬が私の隣で寝ています。　　松村　崇史(23)奈良県

Gate

Takayama Tadanobu 高山 忠信



X-BASICで手軽にカードゲームが作成できるCARDFNCシステムを使ったトランプのひとり遊びです。予備札の使い方が成功の秘訣でしょうか。なお，CARDDRV.X，CARD2.FNCは「秋祭りPRO-68K」にも収録されています。

Gateのルール

このゲームはその名のとおり，カードが門の形に並べられるソリティア（ひとり遊び）の一種です。

では入力方法から解説しましょう。このゲームではCARDDRV.Xというカードゲーム専用のドライバを使用しています。これはOh!X1993年10月号の付録ディスク「秋祭りPRO-68K」のなかに収録されているので，お持ちでない方はそれを使用してください。

あらかじめ，

　A>CARDDRV　TR.DAT

のようにCARDDRVを常駐させておき，CARD2.FNCを組み込んだX-BASICを立ち上げます。あとはそのままリスト1の内容を打ち込んでいってください。

＊　＊　＊

さて，このゲームのルールは「エースアップ」を複雑にしたものとなっているのですが，まず図を見てください。

ゲームを始めると，まず5枚のカードが左右に1列ずつ並べられ，これらは予備札となります。次に，その間に8枚のカードが置かれていきます。これらは場札となります。

話は前後しますが，このゲームの目的は，最上列の台札を置くスペースに（予備札と予備札の間）各スートを数の小さい順に置いていくことです。

そのために，場札・予備札で台札に置けるものがあれば，どんどん台札に移動させてください（Aから順に）。ですが，それだけでは当然のことながら，最大18枚しか移動できません。そこで，右下にある手札をめくっていきます。これは，真上に捨て札として重ねられていきます。

捨て札はトップカードの1枚のみが使え，台札と場札の上に移動させることができます。

なお，手札はなくなると1回のみ捨て札を手札に戻して使うことができますが，それでも難しいと思う方は，変数loop_maxの値を変えてみてください。

また，場札の列には，赤・黒交互で数下がりのシークェンスならばよそのカードを移動させることができます。

例

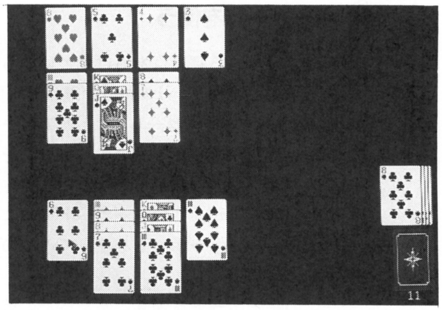
　ハートの6の上に → スペードの5

　クラブのQの上に → ダイヤのJ

これは，赤・黒交互になっている場札の列のトップカードからの複数枚のカード，捨て札のトップカード，台札のトップカード，予備札を移動させてくることができます。

場札の列に空白ができた場合，予備札からどのカードでも移動させてくることができます。

このゲームではマウスの左右ボタンとも同じ動作をします。場札の列を移動させたいときには，その重ねられている中でいちばん下に重ねられているカードをドラッグしてください。

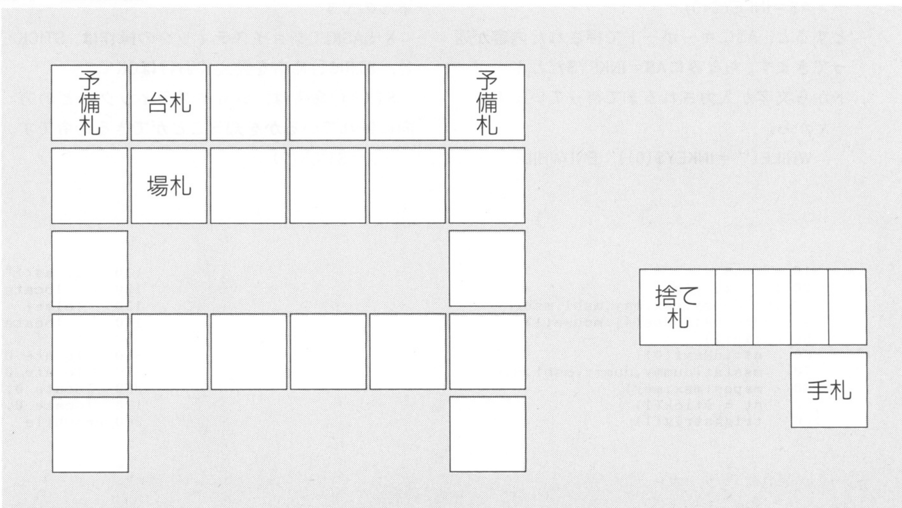


重なった場札の移動について

場札の列にどんどん重ねていくと，7枚を越えたところで途端にカードとカードの間の幅が狭まります。しかし，安心してください。これは表示幅の関係でしかたなしにしていることで，移動には一切支障はきたしません。

とはいうものの，カードを持ってくると

図1　初期配置



きならトップカードははっきり見えているから大丈夫でしょうが，持っていくときには，重ねられたカードがなにかさっぱりわからないことでしょう。でも大丈夫。重ねられたカードのどれかをクリックしてください。するとカードが，がばっと広がります。ボタンから一度指を離したら，表示幅が広いときと同様，ドラッグ操作で移動させてください。

カードの移動の際，もし「この列はや～めた」と思ったら，どこかよその場所をクリックしてください。この場合，もしそこが移動できるカードのところであれば，ボタンから指を離すことなくドラッグ操作に移れます。

言葉で書くと予想以上に長い説明になってしまいましたが，やってみればすぐにわかると思います（この説明はいったいなんだったのだろう……。まあ，カードゲームというものはほとんどがそのパターンなんですがね）。

プログラムについて

このプログラムの構造そのものはいたってオーソドックスに作ってあります。しかし，関数1つひとつをみると異様に入り組んだものもあります。これは実験的試みとして「ローカル変数を使わずにどれだけ関数が組めるか？」なんてことをやってしまったためです。

そのほか，一度画面に表示したものと同じものを表示するときには，できるだけget(), put()を多用して，高速化しています。これも当然プログラムを汚くします。

しかし，これには愛がこもってます。少しでも，プレイヤーのストレスを減らすべく，自分ひとりがしょいこめばいいものならと，ストレスの洗礼を浴び続けた結果なのです。

また，その結果として，プログラムの長さも長くなります。しかし，プレイヤーのストレス度に比較すると，少々大きくなっても十分なメリットがあると思います（反論非希望）。

●参考にした本

桐山雅光，トランプの遊び方，有紀書房

表1　変数表

mx,my,bl,br	……マウス用
loop	……手札を何回使ったか
loop_max	……手札を何回まで使えるか
chk	……何枚台札に置いたか
fin	……終了判定
te_kazu	……手札の数
sute_kazu	……捨て札の数
dat2flag	……dat2()のデータが有効か
card()	……手札の内容
ba(,)	……場札の内容
ba_kazu()	……場札の各列のカードの数
yobi()	……予備札の内容
fnd()	……台札の内容
sute()	……捨て札の内容
dat()	……場札の列を移動するときのget(), put()用のバッファ（間が広いとき用）
dat2()	……場札の列を移動するときのget(), put()用のバッファ（間が狭いとき用）
mxl,myl,my2	……移動させるカードの座標

リスト1

```
  10 /*
  20 /* Gate
  30 /*  Programmed by T.Takayama '93. 5.14(Fri.)-'93. 5.18(Tue.
)
  40 /*
  50 int mx,my,bl,br
  60 int loop,loop_max=2
  70 char chk,fin,te_kazu,sute_kazu
  80 int dat2flag=0
  90 dim char card(51),ba(7,12),ba_kazu(7),yobi(9),fnd(3),sute(3
3),dat(7026),dat2(4488)
 100 /*
 110 int mx1,my1,my2
 120 /*
 130 prep()
 140 repeat
 150   init1()
 160   for loop=1 to loop_max
 170    init2()
 180     while chk<52 and fin
 190       game()
 200     endwhile
 210     if chk=52 then break
 220   next
 230 until replay()
 240 screen 2,0,1,0
 250 mouse(0)
 260 end
 270 /*   1ゲーム毎の初期化
 280 func init1()
 290   int i
 300   mouse(2)
 310   for i=0 to 51
 320     card(i)=i+1
 330   next
 340   shuffle(52)
 350   for i=0 to 3
 360     fnd(i)=0
 370   next
 380   te_kazu=52
 390   sute_kazu=0
 400   wipe()
 410   c_put(462,360,0)
 420   w_te_kazu()
 430   for i=0 to 9
 440   . yobi(i)=card(te_kazu-1)
 450     te_kazu=te_kazu-1
 460     w_te_kazu()
 470     w_yobi(i)
 480     sound(2)
 490   next
 500   for i=0 to 7
 510     ba(i,0)=card(te_kazu-1)
 520     te_kazu=te_kazu-1
 530     w_te_kazu()
 540     ba_kazu(i)=1
 550     w_ba(i,0)
 560     sound(2)
 570   next
 580   chk=0
 590 endfunc
 600 /*   1ループ毎の初期化
 610 func init2()
 620   int i
 630   if 1<loop then {
 640     for i=0 to sute_kazu-1
 650       card(i)=sute(i)
 660     next
 670     shuffle(sute_kazu)
 680     te_kazu=sute_kazu
 690     fill(330,258,508,352,0)
 700     c_put(462,360,0)
 710     w_te_kazu()
 720     sute_kazu=0
 730     move_te_to_sute()
 740   }
 750   fin=1
 760   mouse(1)
 770 endfunc
 780 /*   カードをシャッフルする
 790 func shuffle(a;int)
 800   int i,k,s,t
 810   for i=0 to 99
 820     s=rnd()*a:t=rnd()*a
 830     k=card(s)
 840     card(s)=card(t)
 850     card(t)=k
 860   next
 870 endfunc
 880 /*   プレイヤーの処理
 890 func game()
 900   int p
 910   mson()
 920   p=select()
 930   if  0<=p and p<=7  then {
 940     if ba_kazu(p)<>0 then procedure_ba(p)
 950   }
 960   if  8<=p and p<=17 then {
 970     p=p-8
 980     if yobi(p)<>0 then procedure_yobi(p)
 990   }
1000   if 18<=p and p<=21 then {
1010     p=p-18
1020     if fnd(p)<>0 then procedure_fnd(p)
1030   }
1040   if p=22 then procedure_sute()
1050   if p=23 then move_te_to_sute()
1060   if p=24 then fin=0
1070 endfunc
1080 /*   プレイヤーの指すところ
1090 func select()
1100   int r=-1,x
1110   if area( 59,107,267,501) then {
1120     r=(mx-59)¥54
1130     if 311<=my then r=r+4
1140   }
1150   if area(  5,  5, 51,507) then r=12-(my-5)¥102
1160   if area(275,  5,321,507) then r=17-(my-5)¥102
```

```
1170   if area( 59,  5,267, 99) then r=18+(mx-59)¥54
1180   x=466-sute_kazu*4
1190   if area(  x,258,x+46,352) and 0<sute_kazu then r=22
1200   if area(462,360,508,454) and 0<te_kazu then r=23
1210   if area(462,393,508,422) and te_kazu=0 then r=24
1220   return(r)
1230 endfunc
1240 /*   台札を指した時の処理
1250 func procedure_fnd(a;int)
1260   int f
1270   if mx<=a*54+105 then {
1280     mx1=a*54+59:my1=5:my2=99
1290     f=fnd(a)
1300     pre_moving_one_card(f)
1310     if number(f)=1 then {
1320       fill(mx1,5,mx1+46,99,0)
1330       fnd(a)=0
1340     } else {
1350       fnd(a)=fnd(a)-1
1360       w_fnd(a)
1370     }
1380     if procedure_moving_one_card(f,0) then chk=chk-1 else f
nd(a)=f:w_fnd(a)
1390   }
1400 endfunc
1410 /*   場札を指した時の処理
1420 func procedure_ba(a;int)
1430   int b,c,d,p,f=1
1440   b=ba_kazu(a)-1
1450   mx1=(a mod 4)*54+59
1460   my1=ba_y(a,b):my2=my1+94
1470   if mx<=mx1+46 and my<=my2 then {
1480     if my1<=my then {
1490       pre_moving_one_card(ba(a,b))
1500       fill(mx1,my1,mx1+46,my2,0)
1510       if 0<b then w_ba(a,b-1)
1520       if procedure_moving_one_card(ba(a,b),0) then {
1530         if add_ba_kazu(a,-1) then w_ba_retu(a,0,ba_kazu(a)-
1)
1540       } else w_ba(a,b)
1550     } else {
1560       if b<7 then c=pre_moving_ba_retu_1(a) else c=pre_movi
ng_ba_retu_2(a,b)
1570         if 0<=c then {
1580           move_card()
1590           p=select()
1600           if 0<=p and p<=7 then {
1610             if check_setting_ba(ba(a,c),p) then {
1620               c=c-b
1630               if add_ba_kazu(a,c-1) then w_ba_retu(a,0,ba_kaz
u(a)-1)
1640               for f=0 to 1-c
1650                 ba(p,ba_kazu(p)+f)=ba(a,ba_kazu(a)+f)
1660               next
1670               b=ba_y(p,ba_kazu(p)-1)
1680               if add_ba_kazu(p,1-c) then {
1690                 f=ba_y(p,ba_kazu(p)-1)
1700                 d=(p mod 4)*54+59
1710                 if f<b then fill(d,f+95,d+46,b+94,0)
1720                 if dat2flag then {
1730                   w_ba_retu(p,1,ba_kazu(p)+c-2)
1740                   c=ba_y(p,ba_kazu(p)+c)
1750                   ue_kugiri(d,c-8)
1760                   put(d,c-8,d+46,f+94,dat2)
1770                 } else {
1780                   w_ba_retu(p,1,ba_kazu(p)-1)
1790                 }
1800               }  else {
1810                 if ba_kazu(p)<8 or dat2flag then {
1820                   c=ba_y(p,ba_kazu(p)+c-1)
1830                   b=(p mod 4)*54+59
1840                   if ba_kazu(p)<8 then {
1850                     put(b,c,b+46,ba_y(p,ba_kazu(p)-1)+94,dat)
1860                   } else {
1870                     put(b,c,b+46,ba_y(p,ba_kazu(p)-1)+94,dat2
)
1880                   }
1890                   line(b+1,c-1,b+45,c-1,1)
1900                 } else w_ba_retu(p,ba_kazu(p)+c-1,ba_kazu(p)-
1)
1910               }
1920               m_play(1)
1930               f=0
1940             }
1950           }
1960           if 18<=p and p<=21 and c=b then {
1970             if check_setting_fnd(ba(a,b),p-18) then {
1980               if add_ba_kazu(a,-1) then w_ba_retu(a,0,ba_kazu
(a)-1)
1990               f=0
2000             }
2010           }
2020           if 95<my2-my1 then {
2030             apage(0)
2040             fill(0,95,46,my2-my1,0)
2050             apage(2)
2060           }
2070         } else f=0
2080         if f then {
2090           if b<7 then {
2100             if 0<c then line(mx1+1,my1-1,mx1+45,my1-1,1)
2110             put(mx1,my1,mx1+46,my2,dat)
2120           } else {
2130             b=ba_y(a,c)-1
2140             if 0<c then line(mx1+1,b,mx1+45,b,1)
2150             put(mx1,b+1,mx1+46,ba_y(a,ba_kazu(a))+86,dat2)
2160           }
2170         }
2180         dat2flag=0
2190     }
2200   }
2210 endfunc
2220 /*   場札の列を動かす前処理（列が7枚未満の場合）
2230 func pre_moving_ba_retu_1(a;int)
2240   int r
2250   my1=(a¥4)*207+107
2260   r=(my-my1)¥17
2270   my1=ba_y(a,r)
2280   get(mx1,my1,mx1+46,my2,dat)
2290   pre_move_card()
2300   put(0,0,46,my2-my1,dat)
2310   pset(0,0,0):pset(46,0,0)
2320   vpage(13)
2330   apage(2)
2340   fill(mx1,my1,mx1+46,my2,0)
2350   if 0<r then w_ba(a,r-1)
2360   return(r)
2370 endfunc
2380 /*   場札の列を動かす前処理（列が7枚以上の場合）
2390 func pre_moving_ba_retu_2(a;int,b;int)
2400   int r,i,p
2410   apage(1)
2420   if a<4 then r=107 else r=my1-b*17
2430   for i=0 to b
2440     p=r+i*17
2450     c_put(mx1,p,ba(a,i))
2460     ue_kugiri(mx1,p)
2470   next
2480   p=p+94
2490   if a<4 then {
2500     pset(mx1,p,1):pset(mx1+46,p,1)
2510     line(mx1+1,p+1,mx1+45,p+1,1)
2520   }
2530   vpage(14)
2540   msoff()
2550   mson()
2560   if area(mx1,r,mx1+46,p) then {
2570     i=my2
2580     my1=r:my2=p
2590     if p-95<my then r=ba_kazu(a)-1 else r=(my-r)¥17
2600     my1=my1+r*17
2610     get(mx1,my1,mx1+46,p,dat)
2620     pre_move_card()
2630     put(0,0,46,p-my1,dat)
2640     pset(0,0,0):pset(46,0,0)
2650     apage(2)
2660     p=ba_y(a,r)
2670     dat2flag=1
2680     get(mx1,p,mx1+46,i,dat2)
2690     fill(mx1,p,mx1+46,i,0)
2700     if 0<r then w_ba(a,r-1)
2710     vpage(13)
2720     apage(1)
2730   } else {
2740     vpage(12)
2750     r=-1
2760   }
2770   fill(mx1,107,mx1+46,501,0)
2780   apage(2)
2790   return(r)
2800 endfunc
2810 /*   予備札を指した時の処理
2820 func procedure_yobi(a;int)
2830   mx1=5+(a¥5)*270:my1=413-(a mod 5)*102
2840   my2=my1+94
2850   pre_moving_one_card(yobi(a))
2860   fill(mx1,my1,mx1+46,my2,0)
2870   if procedure_moving_one_card(yobi(a),1) then yobi(a)=0 el
se w_yobi(a)
2880 endfunc
2890 /*   捨て札を指した時の処理
2900 func procedure_sute()
2910   int x
2920   mx1=466-sute_kazu*4:my1=258:my2=352
2930   pre_moving_one_card(sute(sute_kazu-1))
2940   if sute_kazu=1 then x=46 else x=3
2950   fill(mx1,258,mx1+x,352,0)
2960   if 1<sute_kazu then w_sute(sute_kazu-2)
2970   if procedure_moving_one_card(sute(sute_kazu-1),0) then {
2980     sute_kazu=sute_kazu-1
2990     if sute_kazu=0 and te_kazu=0 and loop<loop_max then {
3000       fill(462,393,508,422,6)
3010       symbol(468,396,"END",1,1,2,13,0)
3020       loop=loop_max
3030     }
3040   } else {
3050     w_sute(sute_kazu-1)
3060   }
3070 endfunc
3080 /*   1枚のカードを動かす時の前処理
3090 func pre_moving_one_card(a;int)
3100   pre_move_card()
3110   c_put(0,0,a)
3120   vpage(13)
3130   apage(2)
3140 endfunc
3150 /*   1枚のカードを動かす時の処理
3160 func procedure_moving_one_card(a;int,b;int)
3170   int r=0,p,n
3180   move_card()
3190   p=select()
3200   if  0<=p and p<=8  then {
3210     if check_setting_ba(a,p) or (b and ba_kazu(p)=0) then {
3220       ba(p,ba_kazu(p))=a
3230       if add_ba_kazu(p,1) then {
3240         n=(p¥4)*204+259
3250         r=(p mod 4)*54+59
3260         fill(r,n,r+46,n+44,0)
3270         w_ba_retu(p,1,ba_kazu(p)-1)
3280       } else {
3290         w_ba(p,ba_kazu(p)-1)
3300       }
3310       m_play(1)
3320       r=1
```


▶MacintoshではPowerPC機が出ますが，X68000でも可能性はあるのでしょうかねえ。
沢田　裕一(29)兵庫県

```
3330     }
3340   }
3350   if 18<=p and p<=21 then r=check_setting_fnd(a,p-18)
3360   return(r)
3370 endfunc
3380 /*   指定したカードが場におけるかどうかチェック
3390 func check_setting_ba(a;int,b;int)
3400   int r=0,bb
3410   if 0<ba_kazu(b) then {
3420     bb=ba(b,ba_kazu(b)-1)
3430     r=(number(bb)=number(a)+1) and (RorB(a)<>RorB(bb))
3440   }
3450   return(r)
3460 endfunc
3470 /*   指定したカードが台札におけるかどうチェックして、置けたら置く
3480 func check_setting_fnd(a;int,b;int)
3490   int r=0,n
3500   n=number(a)
3510   if (n=1 and fnd(b)=0) or (a=fnd(b)+1 and n<>1) then {
3520     fnd(b)=a
3530     w_fnd(b)
3540     m_play(1,3)
3550     chk=chk+1
3560     r=1
3570   }
3580   return(r)
3590 endfunc
3600 /*   場札の数に加える
3610 func add_ba_kazu(b;int,p;int)
3620   int b1
3630   b1=ba_kazu(b)
3640   ba_kazu(b)=ba_kazu(b)+p
3650   return((7<b1)<>(7<ba_kazu(b)))
3660 endfunc
3670 /*   手札から捨て札に移す
3680 func move_te_to_sute()
3690   te_kazu=te_kazu-1
3700   if te_kazu=0 then {
3710     fill(462,360,508,471,0)
3720     fill(462,393,508,422,6)
3730     if loop=loop_max or sute_kazu=0 then {
3740       symbol(468,396,"END",1,1,2,13,0)
3750       loop=loop_max
3760     } else {
3770       symbol(462,396,"NEXT",1,1,2,13,0)
3780     }
3790   } else w_te_kazu()
3800   sute(sute_kazu)=card(te_kazu)
3810   w_sute(sute_kazu)
3820   sound(2)
3830   sute_kazu=sute_kazu+1
3840   msoff()
3850 endfunc
3860 /*   台札を描く
3870 func w_fnd(a;int)
3880   c_put(a*54+59,5,fnd(a))
3890 endfunc
3900 /*   場札の列を描く
3910 func w_ba_retu(a;int,s;int,e;int)
3920   int i
3930   for i=s to e
3940     w_ba(a,i)
3950   next
3960 endfunc
3970 /*   場札を描く
3980 func w_ba(a;int,b;int)
3990   int x,y
4000   x=(a mod 4)*54+59
4010   y=ba_y(a,b)
4020   c_put(x,y,ba(a,b))
4030   if 0<b then ue_kugiri(x,y)
4040 endfunc
4050 /*   場札のY座標を求める
4060 func ba_y(a;int,b;int)
4070   int y
4080   y=(a¥4)*204+107
4090   if 7<ba_kazu(a) then y=y+b*8 else y=y+b*17
4100   return(y)
4110 endfunc
4120 /*   予備札を描く
4130 func w_yobi(a;int)
4140   c_put(5+(a¥5)*270,413-(a mod 5)*102,yobi(a))
4150 endfunc
4160 /*   捨て札を描く
4170 func w_sute(a;int)
4180   int x
4190   x=462-4*a
4200   c_put(x,258,sute(a))
4210   if 0<a then {
4220     pset(x+46,258,1)
4230     line(x+47,259,x+47,351,1)
4240     pset(x+46,352,1)
4250   }
4260 endfunc
4270 /*   カードを上方向で区切る
4280 func ue_kugiri(x;int,y;int)
4290   pset(x,y,1):pset(x+46,y,1)
4300   line(x+1,y-1,x+45,y-1,1)
4310 endfunc
4320 /*   手札の数を描く
4330 func w_te_kazu()
4340   str a[2]
4350   a=itoa(te_kazu)
4360   fill(478,456,493,471,0)
4370   symbol(486-len(a)*4,456,a,1,1,1,15,0)
4380 endfunc
4390 /* move_card() の前処理
4400 func pre_move_card()
4410   home(0,512-mx1,512-my1)
4420   apage(0)
4430 endfunc
4440 /*  カードを移動させる
4450 func move_card()
4460   int bx,by,cx,cy,dy,dy2,gx,gy
4470   bx=464+mx-mx1:by=my-my1
4480   cx=48+bx:cy=512+by
4490   dy=my2-my1+1:dy2=cy-dy
4500   repeat
4510     msstat(mx,my,bl,br)
4520     mspos(mx,my)
4530     if bx <mx then gx=47 else gx=cx-mx
4540     if 511<gx then gx=0
4550     if dy2<my then gy=dy else gy=cy-my
4560     if 511<gy then gy=0
4570     home(0,gx,gy)
4580   until (bl+br)=0
4590   vpage(12)
4600 endfunc
4610 /*   数を求める
4620 func number(a;int)
4630   return((a-1) mod 13+1)
4640 endfunc
4650 /*   スートを求める
4660 func suit(a;int)
4670   return((a-1)¥13)
4680 endfunc
4690 /*   カードが赤か黒か
4700 func RorB(a)
4710   return(((a-1)¥13) mod 3=0)
4720 endfunc
4730 /*   マウスが押されるまで待つ
4740 func mson()
4750   repeat
4760     msstat(mx,my,bl,br)
4770   until bl or br
4780   mspos(mx,my)
4790 endfunc
4800 /*   マウスが離されるまで待つ
4810 func msoff()
4820   repeat
4830     msstat(mx,my,bl,br)
4840   until bl+br=0
4850 endfunc
4860 /*   マウスが指定範囲内にあるかどうか?
4870 func area(x1;int,y1;int,x2;int,y2;int)
4880   return((x1<=mx and mx<=x2) and (y1<=my and my<=y2))
4890 endfunc
4900 /*   音を鳴らし、待つ
4910 func sound(a;int)
4920   m_play(a)
4930   repeat
4940   until m_stat(a)=0
4950 endfunc
4960 /*   リプレイ
4970 func replay()
4980   int k
4990   fill(462,393,508,422,0)
5000   apage(1)
5010   if chk=52 then {
5020     symbol(161,363,"Congratulations!",1,1,2, 1,0)
5030     symbol(159,360,"Congratulations!",1,1,2,11,0)
5040   } else {
5050     if 42<chk then k=8 else k=0
5060     box(359-k,475,505,505,7)
5070     fill(360-k,476,504,504,6)
5080     symbol(372-k,482,itoa(52-chk)+"枚 残りました",1,1,1,13,0)
5090   }
5100   fill(198,212,316,302,1)
5110   box(196,210,314,300,5)
5120   box(195,211,313,299,5)
5130   fill(197,212,312,298,4)
5140   symbol(213,227,"Replay?",1,1,2,13,0)
5150   fill(215,267,255,283,3)
5160   fill(263,267,295,283,3)
5170   symbol(224,268,"Yes   No",1,1,1,11,0)
5180   home(0,0,0)
5190   vpage(14)
5200   setmspos(235,275)
5210   msarea(215,267,295,283)
5220   msoff()
5230   mson()
5240   msarea(0,0,511,511)
5250   vpage(12)
5260   fill(159,210,505,505,0)
5270   apage(2)
5280   return(262<mx)
5290 endfunc
5300 /*   準備
5310 func prep()
5320   int i
5330   randomize(val(mid$(time$,4,2)+right$(time$,2)))
5340   screen 1,1,1,1
5350   console ,,0
5360   palet(1,0)
5370   palet(8,rgb(2,14,4))
5380   palet(10,rgb(3,15,5))
5390   mouse(0):mouse(4)
5400   vpage(0)
5410   apage(3)
5420   fill(0,0,511,511,8)
5430   symbol(345,20,"Gate",3,4,2,10,0)
5440   m_init()
5450   for i=1 to 3
5460     m_alloc(i,100):m_assign(i,i)
5470   next
5480   m_trk(1,"q3@45v14t200o2c4")
5490   m_trk(2,"q3@45v14t200o2c24")
5500   m_trk(3,"q2@52v 9t200o4g8")
5510   apage(2)
5520   vpage(12)
5530 endfunc
```

▶１月号には「芸人」として……と書きましたが，結局フリーの手品師となってしまいました。というわけで，あなたの依頼お待ちしています。連絡先は……。

北川　亮(23)東京都

SIDE A

斜めの路面を捉えるための一歩

Tan Akihiko 丹 明彦

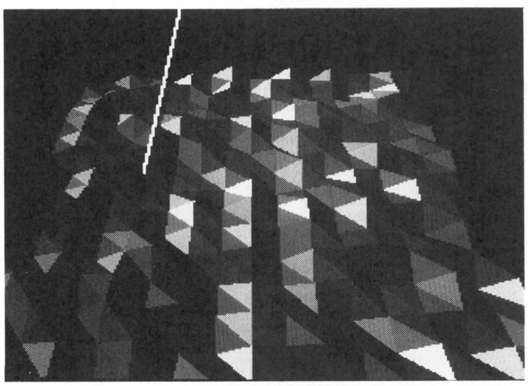
今回は空間内を自由に走り回るために非水平ポリゴンを捉えることを考える
負担をかけず，あらゆるものに対応させるアルゴリズムはあるのだろうか
そこで，すべてに対応できるポリゴンとのヒッティングアルゴリズムを検証する

「バーチャファイター」に寄せて

噂と画面写真でしか見たことのなかったセガの「バーチャファイター」をようやくプレイした。実物を見るまでは，流行の対戦格闘ものと「バーチャレーシング」で培ったポリゴン技術を組み合わせた安易な企画なんじゃないかと思っていたが，それはとんでもない誤解だった。

まず人物の動きが素晴らしい。とにかく実に滑らか。ロボットのようなカクカクした動きではない。パンチやキックを繰り出すときの動き，打撃を叩き込んだときの相手の吹き飛び方，ああ気持ちいい。静と動のメリハリが小気味よく，抑制がきいていながら押さえるところはきっちり押さえてある。とにかくこのゲームは動いているところを見なくては話にならない。

ちょっと通ぶった見かたをするなら，動き，特に打撃のような鋭い動きの表現は3次元CGアニメーションの鉄則をきちんと押さえたものだし，座標系の階層構造が変わるのできわめて表現の難しい投げ技も見事にこなしている。このゲームの実現はひとえにモデリングの職人芸の賜なのである。ポリゴンの人物表現というだけでも難しいのに，格闘家の動きを再現したのだから尋常ではない。



ゲーム自体は，大げさで非科学的な必殺技がなく，硬派な格闘シミュレータの様相を呈している。一般，そしてゲーマーに人気が出るかどうかは未知数だが，このデザインはとても気に入っているのだ。少なくとも3D野郎を引きつけるのは確実。ぜひともヒットすることを願う。

世の中はポリゴンづいているのか，ここで書くネタが結構出てくるものだ。一部のマニアのものでしかなかった3Dポリゴンものが，高速化などの技術的向上と演出力の向上を背景として，一気に一般受けするレベルに高まった感がある。

非水平ポリゴンの空間把握への要件

さて，今回はドライビングシミュレーションの制作はひと休みして，少し回り道をすることにする。「水平でないポリゴンで表現された空間を把握し，その中で運動するためにはなにが必要か」という問題である。むろん，きちんとドライビングシミュレーションをするためには必要なことである。アップダウンやバンクのついた路面をきちんとつかまえるためには，その路面の図形を把握していなくてはならない。

結論からいえば，答えは「いつでもどこでも基底座標系を構築できること」である。加速，旋回，そして減速運動を記述できれば車の運動は表現できるというのは前回も述べたが，これらの運動は基底座標系を用いることで記述できるのだ（と，一応考えている）。

世界の表現方法

3次元CGの世界では，いかに物体があるように「見せる」かということを追求していた。が，ドライビングシミュレータやバーチャルリアリティにお

いては，いかに物体があるように「感じさせる」かということを考えなくてはならない。現実の世界では，そこに机があるなら，手を伸ばせば触れられる。が，3次元CGの世界では，机のオブジェクトと手のオブジェクトを表示したところで終わりである。決して触ることはできない。両者を近づければ，手が机を突き抜けてしまうだろう。これは物体が単なる図形として表現されているからである。

相互に干渉する物体の動きを決める際にはほかの物体の存在を考慮しなくてはならない。一般にCGアニメーションを行うために用いられているキーフレーム法では，これをサポートすることが極めて難しい。

さて，ドライビングシミュレータの舞台となる世界の表現だが，基本的には道路がどこにあるかをきちんと把握できればいいわけだ。

道路といってもいろいろなパターンがある。直線路，カーブ，坂，バンク，交差点，三叉路，ループ，橋，立体交差などなど。個人的に「いつかは鈴鹿」と考えているので，立体交差は外せない。

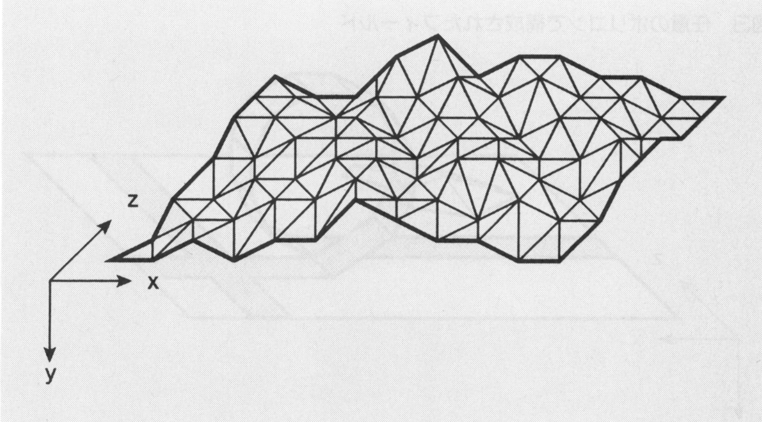
最初は，道を記述するためのデータ構造を考えていたのだが，結局は道を構成するポリゴンとの干渉を計算することにした（カコミ参照）。

1)　ハイトフィールド

一部では2.5次元表現とも呼ばれる。平面座標(x, z)に対応する高さyを関数y=y(x,z)で表現する。壁や天井のように勾配が90°を超えるものは処理できない（天井とは表の面が下を向いているもの。SLASHにおけるポリゴンには表裏がある）。立体交差も処理できない。

ハイトフィールドの効率的な実装方法に，高さyを2次元配列によって表現するものがある（図1）。フィールド全体を三角形要素の規則的なメッシュで等分割し（正則メッシュ），メッシュの各節点にy座標の値を格納する。この方法を使えば，地形を効率的に処理できる。

図1　ハイトフィールド



ポリゴンと平面の方程式

ポリゴンとは3本以上の辺で囲まれた平面である。平面の方程式を復習しておく。

平面は同一直線上にない3点を決定することで決まる。3頂点のポリゴンについて，その平面方程式を求める。4頂点のポリゴンの場合は，その4点が同一平面上にあると仮定して，最初の3点だけを用いる。5頂点以上のポリゴンでも同様にすればよいが，SLASHではポリゴンの頂点数が3または4と決められているので考えないことにする。

平面の方程式は，

$ax+by+cz+d=0$……(1)

で与えられる。ここでベクトル(a,b,c)はその平面の法線ベクトルである。

[証明]

この平面上の任意の2点(x_1,y_1,z_1)，(x_2,y_2,z_2)について式(1)が成立する。すなわち，

$ax_1+by_1+cz_1+d=0$……(1.1)

$ax_2+by_2+cz_2+d=0$……(1.2)

である。ここで式(1.1)-式(1.2)を求めると，

$a(x_1-x_2)+b(y_1-y_2)+c(z_1-z_2)=0$……(1.3)

これは，ベクトル(a,b,c)と$(x_1-x_2,y_1-y_2,z_1-z_2)$の内積が常に0になることを意味する。すなわちベクトル(a,b,c)は平面上の任意のベクトルに垂直である。よってベクトル(a,b,c)は式(1)で表される平面の法線である。

[証明終]

法線の求めかたは，ポリゴンの2本の辺を選んでそれらをベクトルとみなし，外積をとる。SLASHでは頂点が時計回りに並ぶほうが表と規定されているので，

$e_1=(x_2-x_1,y_2-y_1,z_2-z_1)$……(2.1)

$e_2=(x_3-x_2,y_3-y_2,z_3-z_2)$……(2.2)

に対して，

$n=e_1\times e_2=(a,b,c)$……(2.3)

を求めるとよい。残りの係数dは，頂点のひとつを用いて，

$d=-(ax_1+by_1+cz_1)$……(2.4)

のように求める。

ある点が平面の表側にあるか，裏側にあるか，もしくは平面上にあるかは，式(1)にその点の座標を代入してその符号で判別することができる。

$ax+by+cz+d>0$　:　表側にある

$ax+by+cz+d=0$　:　平面上にある

$ax+by+cz+d<0$　:　裏側にある

図　平面の方程式

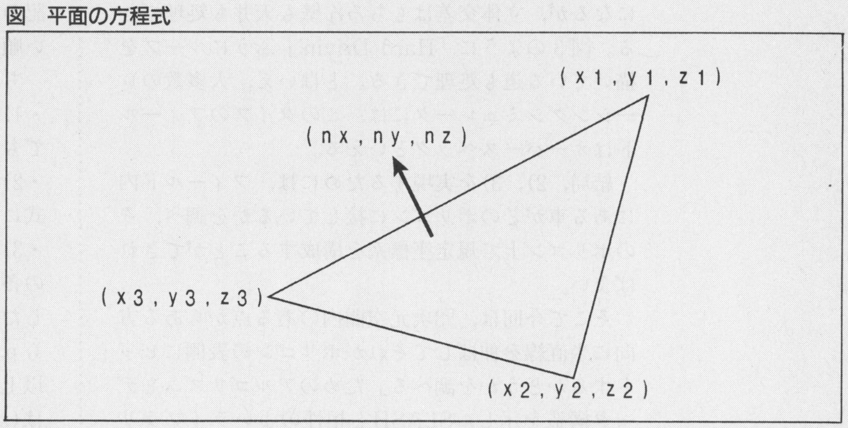


図2　ほぼ上向きのポリゴンで構成されたフィールド

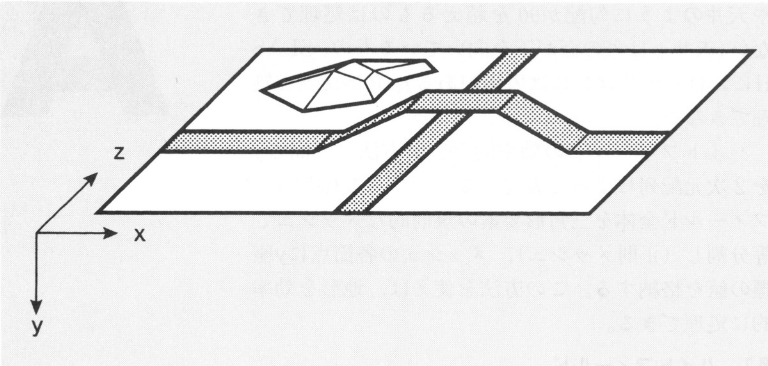
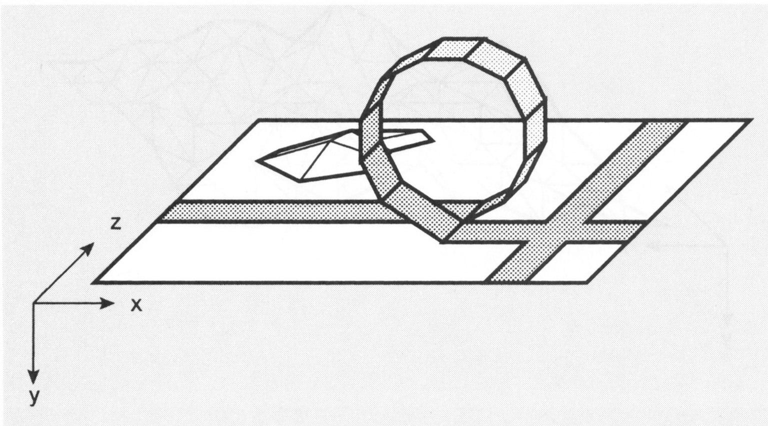


図3　任意のポリゴンで構成されたフィールド



2)　ほぼ上向きのポリゴンで構成されたフィールド

ハイトフィールドから平面座標(x,z)に対応する高さyがひとつに決まるという制限を取り払ったもの(図2)。道路はおおむね上を向いているため，ドライビングシミュレータにはこれで用が足りることが多い。立体交差は処理できるが壁や天井は処理できない。データ構造はハイトフィールドに比べて複雑になる。

3)　任意のポリゴンで構成されたフィールド

一切の制限なし(図3)。データ構造はさらに複雑になるが，立体交差はもちろん壁も天井も処理できる。図3のように「Hard Drivin'」ふうにループを描いている道も処理できる。とはいえ，大多数のレーシングシミュレータには，このタイプのフィールドはオーバースペックといえる。

結局，2)，3)を実現するためには，フィールド内にある車がどのポリゴンに接しているかを調べ，そのポリゴン上で規定座標系を構成することができればよい。

そこで今回は，「3次元空間内のある点からある方向に半直線を伸ばしてそれがポリゴンの表側にヒットするかどうかを調べる」ためのアルゴリズムとデータ構造を示し，SLASHと相性のよいライブラリの形式でプログラムを紹介する。

ヒッティングのアルゴリズム

今回は2つのヒッティングアルゴリズムを紹介するが，そのいずれも，フィールド中に存在するポリゴンは凸多角形であることを前提としている。3項点ポリゴンは常に凸多角形であるから問題ないが，4項点ポリゴンについてはこの点に留意しておく必要がある。

いまさらいうまでもないが，ポリゴンは平面である。平面の性質は簡単なものだが，これを押さえておくと今回のアルゴリズムの理解が容易になるので，コラムに平面の方程式の知識を載せておく。

・ほぼ上向きのポリゴンで構成されたフィールドにおけるヒッティングアルゴリズム(図4)

3次元空間内の点から鉛直下向きに半直線を伸ばし，それがポリゴンと交差するかどうかを調べる。探索起点の位置ベクトルを$\overrightarrow{p}$とし，鉛直下向きベクトルを$\overrightarrow{v}$とする。

「ほぼ上向き」などという回りくどいいい方をした理由は2つある。ひとつは「真上から見るとすべての面が表を向いている」ということが保証されるので計算をさぼれるということ。もうひとつはポリゴンが水平か，またはそれに近ければ車のタイヤの接地点が車軸の真下だと仮定してよくなるということ。

ヒッティングの探索は次のような戦略で行う。

各ポリゴンについて，

1)　$\overrightarrow{p}$がy軸方向から見てポリゴンのバウンディングボックスの内側にあること

2)　$\overrightarrow{p}$がポリゴンの表側にあること

3)　$\overrightarrow{p}$がy軸方向から見てポリゴンの内側にあること

という3つの条件を満たすかどうか調べる。ひとつでも条件を満たさなければ次のポリゴンを調べる。

フィールドにはポリゴンが多数存在するため，上記のアルゴリズムの1)，2)，3)の順序は，計算が軽い順になっている。

ちなみに具体的な算法としては，

・1)のバウンディングボックスのチェックはいうまでもない。

・2)の表側のチェックはコラムに載せた平面の方程式による判別法を用いる。

・3)の内側のチェックは少し複雑である。ポリゴンの各辺をベクトルとみなし，その始点から$\overrightarrow{p}$へ伸ばしたベクトルと外積を取り，その符号がすべて正なら$\overrightarrow{p}$はポリゴンの内側にある。

以上のようになる。y軸方向から見るので，この計算は(x,z)座標のみで行えばよい。また，凸多角形と断ったのは，これが保証されていないと外積によるチ

エックができないからである。

例によって詳しいことはソースコードをご覧いただきたい。

・任意のポリゴンで構成されたフィールドにおけるヒッティングアルゴリズム(図5)

3次元空間内の点から任意の方向に半直線を伸ばし，それがポリゴンと表側から交差するかどうかを調べる。探索起点の位置ベクトルを$\vec{p}$とし，検索方向のベクトルを$\vec{v}$とする。

ヒッティングの探索は次のような戦略で行う。

各ポリゴンについて，

1) $\vec{p}$がポリゴンの表側にあること
2) $\vec{v}$がポリゴンの方向に向いていること
3) $\vec{p}$がポリゴンを$\vec{v}$方向に掃引した立体の内側にあること

という3つの条件を満たすかどうか調べる。ひとつでも条件を満たさなければ次のポリゴンを調べる。

検索方向が任意のため，バウンディングボックスが使えない。このためこのアルゴリズムは計算負荷が著しく上がる。使用するうえでは十分な注意が必要である。

ちなみに具体的な算法としては，

・1)の表側のチェックは，同様に平面方程式による判別法を用いる

・2)の方向のチェックは，$\vec{v}$と平面の法線ベクトルの内積が負であればよい

・3)の内側のチェックは，さらに複雑である。ポリゴンの各辺をベクトルとみなす。辺のベクトル$\vec{e}$と検索方向vの外積$\vec{e}\times\vec{v}$は，その辺をv方向に掃引した平面の法線ベクトル$\vec{n}$になる。$\vec{e}$の始点から$\vec{p}$へ伸ばしたベクトル$\vec{r}$と，ベクトル$\vec{n}$の内積を取り，その符号が正なら$\vec{p}$はこの面に関して掃引体の内側にある。これをポリゴンのすべての辺に対してチェックすればよい。

このアルゴリズムもポリゴンが凸多角形であることが保証されている必要がある。

データ構造

データ構造といってもそれほど大げさなものではなく，SLASHのポリゴンリストをベースにして，CHECKINFO型構造体とCHECKINFOLIST型構造体を定義している。平面方程式の係数(法線ベクトル)やバウンディングボックスなど，形状を定義した時点で計算できるものはできるだけ計算して格納するようにしている。

図4　図2のタイプのフィールドにおける判定

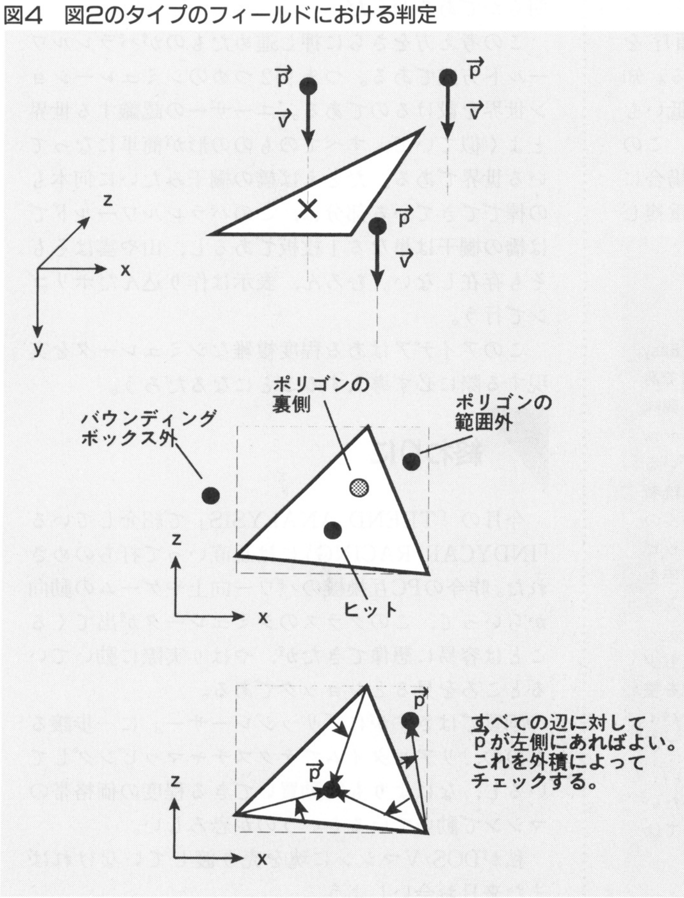
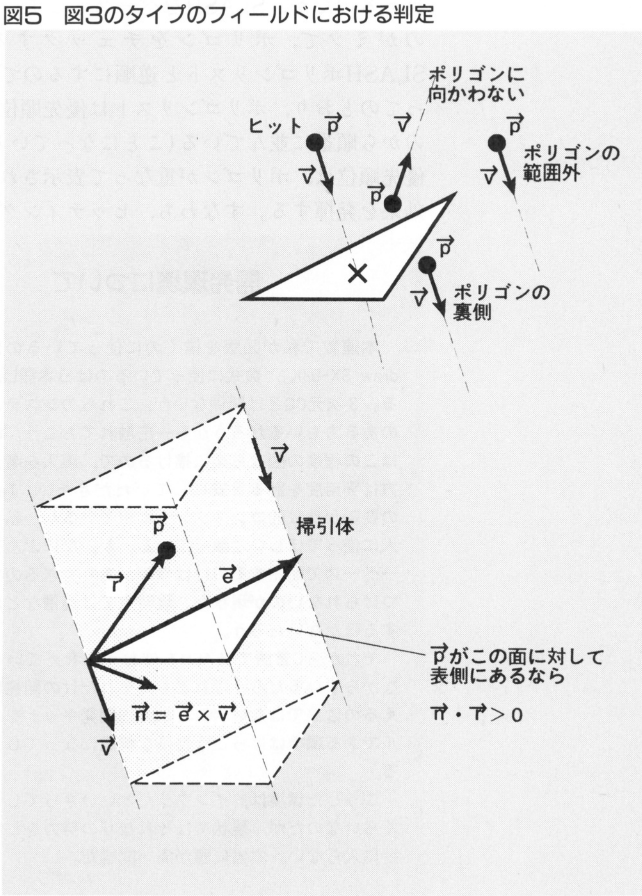


図5　図3のタイプのフィールドにおける判定



SLASHにならってリストと命名しているが，リスト構造ではない。

効率的なヒッティングに関する考察

このヒッティング技術はリアルタイムのシミュレータに応用するわけであるから，できるだけ高速に行わなくてはならない。

高速化といっても例によっていろいろなレベルがある。コーディングを煮つめ，クリティカルな部分はアセンブラで書き直すのは当然だが，ここではもう少し上のレベルの話をする。

まず，ヒッティングであるから，検索起点から走査していってもっとも近くでヒットしたポリゴンを用いるのは当然である。が，上記のアルゴリズムはこれを保証しない。わかるのは個々のポリゴンがヒットしたかしないかだけである。というわけで，ヒットした点までの距離を覚えておいて比較するという方法もあるが，これだと毎回すべてのポリゴンをチェックしなくてはならない。

そこで私の取った方法は，ポリゴン1枚でもヒットしたら即打ち切るという方法である。これだと，もっとも近いポリゴンを見逃す可能性があるように見えるのだが，SLASHをベースとしているというのがミソで，ポリゴンをチェックする順序をSLASHポリゴンリストと逆順にするのである。知ってのとおり，ポリゴンリストは優先順位の低いものから順番に並んでいる(ことになっている)。この優先順位は，ポリゴンが重なって表示される場合に効果を発揮する。すなわち，ヒッティングで重複してヒットする場合にもそのまま適用できる。

これにより，無駄な検索を省き，なおかつ正確なヒッティングも可能になるわけである。

次は無関係のポリゴンをどうやってチェックしないようにするかである。これはブロック化によって劇的な効率向上が図れる。

ブロック化とはSLASHの次期バージョンでマップシステムとして導入されるもので，フィールドをいくつかの大きなポリゴン群(ブロック)に分割しておき，各々のポリゴンの座標変換や表示を行う前に，ブロック単位で表示するかどうかのチェックを行うものである。これにより，明らかに視野に入っていない大多数のポリゴンを処理しなくてもよくなるのである。

で，これはヒッティングのチェックにも利用できるのである。車が現在どのブロックにいるかは簡単に知ることができるので，チェックはそのブロック内のポリゴンに対してのみ行えばよい。

次は，チェックするポリゴンそのものを減らす手口について。簡単にいえば，CHECKINFOLISTには車が通りそうな部分のポリゴンだけを登録すればいいのである。レーシングカーが，山のてっぺんや雲との接触判定をする必要がないというのはほとんど明らかであるからだ。

この考え方をさらに押し進めたものがパラレルワールド方式である。つまり2つめのシミュレーション世界を設けるのである。ユーザーの認識する世界とよく似ていて，すべてのものの形が簡単になっている世界である。たとえば橋の欄干みたいに何本もの棒でできている部分は，このパラレルワールドでは橋の欄干は単なる1枚板であるし，山や雲はそもそも存在しない。むろん，表示は作り込んだポリゴンで行う。

このアイデアはある程度複雑なシミュレータを実現する際に必ず導入することになるだろう。

終わりに

今月の「TREND ANALYSIS」で紹介している「INDYCAR RACING」には正直いって打ちのめされた。昨今のPC互換機のパワー向上やゲームの動向からいって，このクラスのシミュレータが出てくることは容易に想像できたが，やはり実際に動いているところを見るとショックである。

画質ではさすがに「リッジレーサー」に一歩譲るものの，リアルタイムでテクスチャマッピングしているし，なによりも衝動買いできる程度の価格帯のマシンで動いているというのが恐ろしい。

私がDOS/Vマシンに魂を売り渡していなければまた来月お会いしよう。

開発環境について

本連載で私が図版を描くのに使っているのは「Easydraw SX-68K」，数式に使っているのは日本語LaTeXである。3次元CGとは関係ないが，これらのシステムに興味のある方もいるだろうから一応触れておこう。Easydrawはこの程度の図なら楽に描けるので，導入を考えている方は実用度を計る目安にしていただきたい。TeXは読者の要望が比較的コンスタントに出ているし，私も多くの人に使ってほしいと願っている。が，なにぶんフロッピーベースで配布するのには無理がありすぎるので，手をつけられないのが現状だ。現時点では通信などから入手するほかないだろう。

それからC言語でSLASHを使おうと考えている人も少なからずいるだろうけれども，それだけの開発環境を整えるのは楽ではない。特に，SLASH開発キットをコンパイルできる環境はけっこう特殊な構成になってしまっている。

こうした環境は，インフラだといいきってしまいたいくらいなのだが，現状ではそれなりの努力をしなくては手に入らない。本当に頭が痛い問題だ。

■リスト1　MAKEFILE

```
SHELL = a:¥command.x
SLASHLIBDIR = ..¥lib
COLORDIR = ..¥color
CCOPTS = -O -Wall
#CCOPTS = -g -O -Wall
#CCOPTS = -fno-defer-pop -g -O -Wall

%.o: %.s
	has -u -w $<

%.o: %.c
	gcc $(CCOPTS) -c $<

all: ctest.x

ctest.x: ctest.o checklib.o
	gcc ctest.o checklib.o¥
			$(COLORDIR)¥tpllib.a¥
			$(SLASHLIBDIR)¥_slashlib.a¥
			$(SLASHLIBDIR)¥_utillib.a¥
			$(SLASHLIBDIR)¥slashlib.a
#			$(SLASHLIBDIR)¥slashRlib.a¥

ctest.o: ctest.c checklib.h
checklib.o: checklib.c checklib.h

clean:
	if exist *.dat del -y *.dat
	if exist *.bak del -y *.bak
	if exist checklib.o del checklib.o
	if exist ctest.o del ctest.o

distclean: clean
	if exist ctest.x del ctest.x
```

■リスト2　CHECKLIB.H

```
/*
	checklib.h
	- _slashlibの補助関数(簡単な干渉チェック)
	  Nov. 1993	丹 明彦(Oh!X)
*/

#ifndef __CHECKLIB_H__
#define __CHECKLIB_H__

#include	"..¥lib¥_slashlib.h"

/* バウンディングボックスの初期値に使うでかい整数 */
#define HUGEINT (1 << 24)

typedef struct {
	int	i;	/* ポリゴン番号 */

	int	a;	/* 平面方程式の係数 */
	int	b;
	int	c;
	int	d;
	int	abc;	/* a^2 + b^2 + c^2 */
	int	abc2;	/* sqrt( a^2 + b^2 + c^2 ) */

	int	xmin;	/* バウンディングボックス */
	int	ymin;
	int	zmin;
	int	xmax;
	int	ymax;
	int	zmax;

	int	n;	/* 頂点数 */

	int	x[4];	/* 頂点の座標 */
	int	y[4];
	int	z[4];

	int	ex[4];	/* 辺のベクトル */
	int	ey[4];
	int	ez[4];
} CHECKINFO;

typedef struct {
	SLPOLYGONLIST*	pll;	/* 母体ポリゴンリスト */
	SLPOINTLIST*	ptl;	/* 母体ポイントリスト */
	int		xmin;	/* バウンディングボックス */
	int		ymin;
	int		zmin;
	int		xmax;
	int		ymax;
	int		zmax;
	int		n;	/* CHECKINFO数 */
	CHECKINFO	ci[0];
} CHECKINFOLIST;

/* 関数 */
CHECKINFOLIST	*createCheckInfoList( int, SLPOLYGONLIST*, SLPOINTLIST* );
void		destroyCheckInfoList( CHECKINFOLIST* );
int		addCheckInfo( CHECKINFOLIST*, int );
void		generateCheckInfoList( CHECKINFOLIST* );
int		check1( int*,int*, int*, int*, CHECKINFO*, int, int, int, int, int, int );
int		check2( int*, int*, CHECKINFO*, int, int, int );
int		checks1( int*,int*, int*, int*, CHECKINFOLIST*, int, int, int, int, int, int );
int		checks2( int*, int*, CHECKINFOLIST*, int, int, int );

/* 表裏判別式 */
#define DETERMINE(CI,X,Y,Z)	(((CI).a)*(X)+((CI).b)*(Y)+((CI).c)*(Z)+((CI).d))
/* 点と平面の距離(符号つき) */
#define DISTANCE(CI,X,Y,Z)	((((CI).a)*(X)+((CI).b)*(Y)+((CI).c)*(Z)+((CI).d))/((CI).abc2))

#endif /* __CHECKLIB_H__ */
```

■リスト3　CHECKLIB.C

```
/*
  checklib.c
  - _slashlibの補助関数(簡単な干渉チェック)
    Nov. 1993  丹 明彦(Oh!X)
*/

#include  <stdlib.h>
#include  "checklib.h"

CHECKINFOLIST  *createCheckInfoList( n, pll, ptl )
  int    n;
  SLPOLYGONLIST  *pll;
  SLPOINTLIST  *ptl;
{
  CHECKINFOLIST  *cil;

  cil = (CHECKINFOLIST*)malloc( sizeof(CHECKINFOLIST)+sizeof(CHECKINFO)*n );
  if ( cil != (CHECKINFOLIST*)0 ) {
    cil->n = 0;
    cil->pll = pll;
    cil->ptl = ptl;
    cil->xmin = HUGEINT;
    cil->ymin = HUGEINT;
    cil->zmin = HUGEINT;
    cil->xmax = -HUGEINT;
    cil->ymax = -HUGEINT;
    cil->zmax = -HUGEINT;
  }
  return cil;
}

void  destroyCheckInfoList( cil )
  CHECKINFOLIST  *cil;
{
  free( cil );
  return;
}

int  addCheckInfo( cil, i )
  CHECKINFOLIST  *cil;
  int    i;
{
  int    n;
  CHECKINFO  *ci;
  SLPOLYGONLIST  *pll;
  SLPOINTLIST  *ptl;
  int    j;

  n = cil->n;
  ci = cil->ci;
  pll = cil->pll;
  ptl = cil->ptl;

  /* ポリゴンの種類を調べて頂点と辺を登録する */
  switch ( pll->polygon[i].type ) {
  case SLPOLYGON_TRIANGLE:
    ci[n].n = 3;
    ci[n].x[0] = ptl->point[pll->polygon[i].v1].x;
    ci[n].y[0] = ptl->point[pll->polygon[i].v1].y;
    ci[n].z[0] = ptl->point[pll->polygon[i].v1].z;

    ci[n].x[1] = ptl->point[pll->polygon[i].v2].x;
    ci[n].y[1] = ptl->point[pll->polygon[i].v2].y;
    ci[n].z[1] = ptl->point[pll->polygon[i].v2].z;

    ci[n].x[2] = ptl->point[pll->polygon[i].v3].x;
    ci[n].y[2] = ptl->point[pll->polygon[i].v3].y;
    ci[n].z[2] = ptl->point[pll->polygon[i].v3].z;

    ci[n].ex[0] = ci[n].x[1] - ci[n].x[0];
    ci[n].ey[0] = ci[n].y[1] - ci[n].y[0];
    ci[n].ez[0] = ci[n].z[1] - ci[n].z[0];

    ci[n].ex[1] = ci[n].x[2] - ci[n].x[1];
    ci[n].ey[1] = ci[n].y[2] - ci[n].y[1];
    ci[n].ez[1] = ci[n].z[2] - ci[n].z[1];

    ci[n].ex[2] = ci[n].x[0] - ci[n].x[2];
    ci[n].ey[2] = ci[n].y[0] - ci[n].y[2];
    ci[n].ez[2] = ci[n].z[0] - ci[n].z[2];
    break;
  case SLPOLYGON_TETRAGON:
    ci[n].n = 4;
    ci[n].x[0] = ptl->point[pll->polygon[i].v1].x;
    ci[n].y[0] = ptl->point[pll->polygon[i].v1].y;
    ci[n].z[0] = ptl->point[pll->polygon[i].v1].z;

    ci[n].x[1] = ptl->point[pll->polygon[i].v2].x;
    ci[n].y[1] = ptl->point[pll->polygon[i].v2].y;
    ci[n].z[1] = ptl->point[pll->polygon[i].v2].z;

    ci[n].x[2] = ptl->point[pll->polygon[i].v3].x;
    ci[n].y[2] = ptl->point[pll->polygon[i].v3].y;
    ci[n].z[2] = ptl->point[pll->polygon[i].v3].z;

    ci[n].x[3] = ptl->point[pll->polygon[i].v4].x;
    ci[n].y[3] = ptl->point[pll->polygon[i].v4].y;
    ci[n].z[3] = ptl->point[pll->polygon[i].v4].z;

    ci[n].ex[0] = ci[n].x[1] - ci[n].x[0];
    ci[n].ey[0] = ci[n].y[1] - ci[n].y[0];
    ci[n].ez[0] = ci[n].z[1] - ci[n].z[0];

```

```
    ci[n].ex[1] = ci[n].x[2] - ci[n].x[1];
    ci[n].ey[1] = ci[n].y[2] - ci[n].y[1];
    ci[n].ez[1] = ci[n].z[2] - ci[n].z[1];

    ci[n].ex[2] = ci[n].x[3] - ci[n].x[2];
    ci[n].ey[2] = ci[n].y[3] - ci[n].y[2];
    ci[n].ez[2] = ci[n].z[3] - ci[n].z[2];

    ci[n].ex[3] = ci[n].x[0] - ci[n].x[3];
    ci[n].ey[3] = ci[n].y[0] - ci[n].y[3];
    ci[n].ez[3] = ci[n].z[0] - ci[n].z[3];
    break;
  default:  /* 線分や点は干渉判定の対象にしないので登録しない */
    return -1;
    break;
  }
  /* ポリゴン番号 */
  ci[n].i = i;
  /* バウンディングボックス */
  ci[n].xmin = HUGEINT;
  ci[n].ymin = HUGEINT;
  ci[n].zmin = HUGEINT;
  ci[n].xmax = -HUGEINT;
  ci[n].ymax = -HUGEINT;
  ci[n].zmax = -HUGEINT;
  for ( j = 0; j < ci[n].n; j++ ) {
    if ( ci[n].x[j] < ci[n].xmin ) ci[n].xmin = ci[n].x[j];
    if ( ci[n].y[j] < ci[n].ymin ) ci[n].ymin = ci[n].y[j];
    if ( ci[n].z[j] < ci[n].zmin ) ci[n].zmin = ci[n].z[j];
    if ( ci[n].x[j] > ci[n].xmax ) ci[n].xmax = ci[n].x[j];
    if ( ci[n].y[j] > ci[n].ymax ) ci[n].ymax = ci[n].y[j];
    if ( ci[n].z[j] > ci[n].zmax ) ci[n].zmax = ci[n].z[j];
  }
  /* リスト全体のバウンディングボックス */
  if ( ci[n].xmin < cil->xmin ) cil->xmin = ci[n].xmin;
  if ( ci[n].ymin < cil->ymin ) cil->ymin = ci[n].ymin;
  if ( ci[n].zmin < cil->zmin ) cil->zmin = ci[n].zmin;
  if ( ci[n].xmax > cil->xmax ) cil->xmax = ci[n].xmax;
  if ( ci[n].ymax > cil->ymax ) cil->ymax = ci[n].ymax;
  if ( ci[n].zmax > cil->zmax ) cil->zmax = ci[n].zmax;
  /* 平面方程式の係数(a,b,c)=法線ベクトル */
  /* 法線ベクトル=e1×e0 */
  ci[n].a = ci[n].ey[1] * ci[n].ez[0] - ci[n].ez[1] * ci[n].ey[0];
  ci[n].b = ci[n].ez[1] * ci[n].ex[0] - ci[n].ex[1] * ci[n].ez[0];
  ci[n].c = ci[n].ex[1] * ci[n].ey[0] - ci[n].ey[1] * ci[n].ex[0];
  /* abcはエッジ長さの4乗のオーダーになるので
     大きなポリゴンに対するa,b,cを小さくする */
#define  TH  8191
  while ( ci[n].a > TH || ci[n].a < -TH ||
    ci[n].b > TH || ci[n].b < -TH ||
    ci[n].c > TH || ci[n].c < -TH ) {
      ci[n].a /= 2;
      ci[n].b /= 2;
      ci[n].c /= 2;
    }
  /* 平面方程式の係数d=-(ax+by+cz) */
  ci[n].d = -( ci[n].a * ci[n].x[0] + ci[n].b * ci[n].y[0] + ci[n].c * ci[n].z[0] );
  /* その他のパラメータ */
  ci[n].abc = ci[n].a * ci[n].a + ci[n].b * ci[n].b +  ci[n].c * ci[n].c;
  ci[n].abc2 = sqrI( ci[n].abc );
  /* 登録 */
  cil->n = n+1;
  /* 登録したインデックスを返す */
  return n;
}

void  generateCheckInfoList( cil )
  CHECKINFOLIST  *cil;
{
  int  i;
  for ( i = 0; i < (cil->pll)->n; i++ )
    addCheckInfo( cil, i );
  return;
}

/* 汎用の交差判定 */
/* 戻り値はリザルトコード */
int  check1( rx, ry, rz, rd, ci, px, py, pz, vx, vy, vz )
  int     *rx, *ry, *rz;  /* 結果(交点) */
  int     *rd;      /* 結果(距離) */
  CHECKINFO  *ci;
  int     px, py, pz;  /* 検索起点 */
  int     vx, vy, vz;  /* 検索方向 */
{
  int  d, i, v;

  /* 検索起点はポリゴンの表側にある必要がある */
  d = DETERMINE( *ci, px, py, pz );
  if ( d < 0 ) return 0;
  /* 検索方向はポリゴンに向かう必要がある */
  if ( (ci->a * vx + ci->b * vy + ci->c * vz) >= 0 ) return 0;
  /* 検索起点が"ポリゴンを検索方向に掃引した立体"の内側かどうか調べる */
  for ( i = 0; i < ci->n; i++ ) {
    if ( (((ci->ez[i] * vy) - (ci->ey[i] * vz)) * (px - ci->x[i]) +
          ((ci->ex[i] * vz) - (ci->ez[i] * vx)) * (py - ci->y[i]) +
          ((ci->ey[i] * vx) - (ci->ex[i] * vy)) * (pz - ci->z[i])) < 0 )
        return 0;
  }
  /* 距離 */
  v = (ci->a * vx + ci->b * vy + ci->c * vz);
  *rd = -d/v;
  /* 交点を求める */
  *rx = px + vx*(-d)/v;
  *ry = py + vy*(-d)/v;
  *rz = pz + vz*(-d)/v;
  return 1;
}

/* zx平面にほぼ平行と仮定してよい上向きポリゴンの交差判定 */
/* 戻り値はリザルトコード */
int  check2( ry, rd, ci, px, py, pz )
  int     *ry;     /* 結果(交点) */
  int     *rd;     /* 結果(距離) */
  CHECKINFO  *ci;
  int     px, py, pz;  /* 検索起点 */
  /* 検索方向は鉛直下向き(y軸正方向) */
{
  int  d, i;

  /* バウンディングボックスによるチェック */
  if ( px < ci->xmin ) return 0;
  if ( px > ci->xmax ) return 0;
  if ( pz < ci->zmin ) return 0;
  if ( pz > ci->zmax ) return 0;
  /* 検索起点はポリゴンの表側にある必要がある */
  d = DETERMINE( *ci, px, py, pz );
  if ( d < 0 ) return 0;
  /* y軸方向からみて検索起点がポリゴンの内側かどうか調べる */
  for ( i = 0; i < ci->n; i++ ) {
    if ( ((pz - ci->z[i]) * ci->ex[i] - (px - ci->x[i]) * ci->ez[i]) > 0 )
      return 0;
  }
  /* 交点を求める */
  *ry = -((ci->a)*px + (ci->c)*pz + (ci->d))/(ci->b);
  /* 距離 */
  /* 計算が重複しているのでDISTANCEマクロは使わない */
  *rd = d/(ci->abc2);
  return 1;
}

/* 汎用の交差判定 */
/* 戻り値はリスト中のインデックス */
int  checks1( rx, ry, rz, rd, cil, px, py, pz, vx, vy, vz )
  int     *rx, *ry, *rz;  /* 結果(交点) */
  int     *rd;      /* 結果(距離) */
  CHECKINFOLIST  *cil;
  int     px, py, pz;  /* 検索起点 */
  int     vx, vy, vz;  /* 検索方向 */
{
  int  i;
  for ( i = ((cil->n)-1); i >= 0; i-- ) {
    if ( check1( rx, ry, rz, rd, &(cil->ci[i]), px, py, pz, vx, vy, vz ) )
      return i;
  }
  return -1;
}

/* zx平面にほぼ平行と仮定してよい上向きポリゴンの交差判定 */
/* 戻り値はリスト中のインデックス */
int  checks2( ry, rd, cil, px, py, pz )
  int     *ry;     /* 結果(交点) */
  int     *rd;     /* 結果(距離) */
  CHECKINFOLIST  *cil;
  int     px, py, pz;  /* 検索起点 */
  /* 検索方向は鉛直下向き */
{
  int  i;
  for ( i = ((cil->n)-1); i >= 0; i-- ) {
    if ( check2( ry, rd, &(cil->ci[i]), px, py, pz ) )
      return i;
  }
  return -1;
}
```

■リスト4 CTEST.C

```
/*
  ctest.c
  - checklibのテストプログラム
    Nov. 1993  丹 明彦(Oh!X)
*/

#define    __IOCS_INLINE__
#include  <iocslib.h>
#define    __DOS_INLINE__
#include  <doslib.h>
#include  <stdio.h>
#include  <stdlib.h>

#include  "..¥lib¥_slashlib.h"
#include  "..¥color¥_tpllib.h"
#include  "checklib.h"

SLPOLYGONLIST  *ground_polygonlist;
SLPOINTLIST  *ground_pointlist;
SLPOLYGONLIST  *line_polygonlist;
SLPOINTLIST  *line_pointlist;
SLTRANSWORK  *work;
SLMINMAX  *minmax1, *minmax2, *minmaxt;
SLPARAMETER  parameter;
CHECKINFOLIST  *ground_checkinfolist;

#define  MESHU  16
#define  MESHV  16

void  setup_ground()
{
  int  i, j;
  static int  x[MESHU+1][MESHV+1], y[MESHU+1][MESHV+1], z[MESHU+1][MESHV+1];

  ground_polygonlist->n = 0;
  ground_pointlist->n = 0;
  for ( i = 0; i < MESHV+1; i++ ) {
    for ( j = 0; j < MESHU+1; j++ ) {
      x[i][j] = (j-MESHU/2)*(1024/MESHU);
      y[i][j] = -rand()/(29*16);
      z[i][j] = (i-MESHV/2)*(1024/MESHV);
    }
  }
  for ( i = 0; i < MESHV; i++ ) {
    for ( j = 0; j < MESHU; j++ ) {
      addtriangle( ground_polygonlist, ground_pointlist,
        x[i][j+1], y[i][j+1], z[i][j+1],
        x[i][j], y[i][j], z[i][j],
        x[i+1][j], y[i+1][j], z[i+1][j],
        &yellow );
      addtriangle( ground_polygonlist, ground_pointlist,
        x[i][j+1], y[i][j+1], z[i][j+1],
        x[i+1][j], y[i+1][j], z[i+1][j],
        x[i+1][j+1], y[i+1][j+1], z[i+1][j+1],
        &red );
```

▶最近，サードパーティのX68000用ハードウェアが増えているのがうれしい。
鈴木　康夫(26)岐阜県

```
      }
    }
    return;
  }

  int  main()
  {
    int  sp, time = 0;
    int  mode = 0;
    int  mscur, x, y, msdt, lb, rb, z = 1000;
    int  shift = 0;
    int  line;
    int  px = 0, py = -800, pz = 0;
    int  vx = 0, vy = -10, vz = 0;
    int  rx = 0, ry = 0, rz = 0, rd;
    int  hl = 0, hltmp, rytmp;
    SLPALET  *hlp = &yellow;

    ground_polygonlist = malloc( sizeof(SLPOLYGONLIST)+sizeof(SL
POLYGON)*(MESHU*MESHV*2) );
    ground_pointlist = malloc( sizeof(SLPOINTLIST)+sizeof(SLPOIN
T)*((MESHU+1)*(MESHV+1)) );
    setup_ground();
    AddNorm( ground_polygonlist, ground_pointlist );

    ground_checkinfolist = createCheckInfoList( MESHU*MESHV*2, g
round_polygonlist, ground_pointlist );
    generateCheckInfoList( ground_checkinfolist );

    line_polygonlist = malloc( sizeof(SLPOLYGONLIST)+sizeof(SLPO
LYGON)*(1) );
    line_pointlist = malloc( sizeof(SLPOINTLIST)+sizeof(SLPOINT)
*(3) );
    line_polygonlist->n = 0;
    line_pointlist->n = 0;
    line = addline( line_polygonlist, line_pointlist,
           rx,   ry, rz,
           rx, -800, rz,
       rx-800, -400, rz,
        &std_white );
    noshade( line_polygonlist, line, RGBILONG(31,31,31,0) );

    work = malloc( sizeof(SLTRANSWORK)*(MESHU+1)*(MESHV+1) );  /
* number of points */
    minmax1 = malloc( sizeof(SLMINMAX)*(2+1) );  /* number of ob
jects + 1 */
    minmax2 = malloc( sizeof(SLMINMAX)*(2+1) );  /* number of ob
jects + 1 */

    CRTMOD( 14 );
    G_CLR_ON();
    B_CUROFF();

    MS_INIT();
    MS_CUROF();
    MS_LIMIT(0,0,255,255);
    MS_CURST(128,128);
    MS_CUROF();
    SKEY_MOD(0,0,0);

    SetClearColor( 0x0000 );
    SetWindowSize( 256, 256 );
    SetWindowCenter( 128, 128 );

    parameter.x = 0;
    parameter.y = 0;
    parameter.z = 0;
    parameter.pitch = 4096-512;
    parameter.head = 2048;
    parameter.bank = 0;
    parameter.alpha = 8;
    parameter.beta = 8;

    sp = SUPER( 0 );

    for (;;) {
      mscur = MS_CURGT();
      x = mscur/65536;
      y = mscur%65536;
      msdt = MS_GETDT()%65536;
      lb = msdt/256;
      rb = msdt%256;

      if ( BITSNS(0x00)&2 ) {  /* ESCキーで終了 */
        while ( BITSNS(0x00)&2 );
        break;
      }

      if ( BITSNS(0x0C)&8 ) {    /* F1: mode切り換え */
        while ( BITSNS(0x0C)&8 );
        mode = 1 - mode;
      }

      if ( BITSNS(0x0E)&1 ) {  /* SHIFTキー併用で視点設定 */
        if ( shift == 0 ) {  /* SHIFTキーが押された瞬間 */
          x = 255-(parameter.head)/16;
          y = 255-(parameter.pitch)/16;
          MS_CURST( x, y );
          shift = 1;
        } else {
          parameter.head = (255-x)*16;
          parameter.pitch = (255-y)*16;
        }
      } else {  /* SHIFTキーなし */
  if ( mode == 0 ) {  /* 任意方向の検索 */
    if ( shift == 1 ) {  /* SHIFTキーが離された瞬間 */
        x = -vx+128;
        y = vz+128;;
        MS_CURST( x, y );
        shift = 0;
      } else {
        ground_polygonlist->polygon[hl].palet = hlp;
        vx = -(x-128);
        vy = 64;
        vz = (y-128);
        hltmp = checks1( &rx, &ry, &rz, &rd, ground_checkinfolist,
 px, py, pz, vx, vy, vz );
        if ( hltmp != -1 ) {
          hl = ground_checkinfolist->ci[hltmp].i;
        } else {
          hl = 0;
          rx = px + vx*8;
          ry = py + vy*8;
          rz = pz + vz*8;
        }
        if ( !lb ) {
          rx = px + vx*8;
          ry = py + vy*8;
          rz = pz + vz*8;
        }
        line_pointlist->point[line_polygonlist->polygon[0].v1].x =
 rx;
        line_pointlist->point[line_polygonlist->polygon[0].v1].y =
 ry;
        line_pointlist->point[line_polygonlist->polygon[0].v1].z =
 rz;
        line_pointlist->point[line_polygonlist->polygon[0].v2].x =
 px;
        line_pointlist->point[line_polygonlist->polygon[0].v2].y =
 py;
        line_pointlist->point[line_polygonlist->polygon[0].v2].z =
 pz;
        line_pointlist->point[line_polygonlist->polygon[0].v3].x =
 px-800;
        line_pointlist->point[line_polygonlist->polygon[0].v3].y =
 py;
        line_pointlist->point[line_polygonlist->polygon[0].v3].z =
 pz;
        hlp = ground_polygonlist->polygon[hl].palet;
        if ( lb )
          ground_polygonlist->polygon[hl].palet = &std_cyan;
      }
  } else {  /* 垂直方向の検索 */
      if ( shift == 1 ) {  /* SHIFTキーが離された瞬間 */
        x = -rx/8+128;
        y = rz/8+128;;
        MS_CURST( x, y );
        shift = 0;
      } else {
        ground_polygonlist->polygon[hl].palet = hlp;
        rx = -(x-128)*8;
        rz = (y-128)*8;
        hltmp = checks2( &rytmp, &rd, ground_checkinfolist, rx, -8
00, rz );
        if ( hltmp != -1 ) {
          hl = ground_checkinfolist->ci[hltmp].i;
          ry = rytmp;
        } else {
          hl = 0;
          ry = 0;
        }
        if ( !lb ) {
          ry = -400;
        }
        line_pointlist->point[line_polygonlist->polygon[0].v1].x =
 rx;
        line_pointlist->point[line_polygonlist->polygon[0].v1].y =
 ry;
        line_pointlist->point[line_polygonlist->polygon[0].v1].z =
 rz;
        line_pointlist->point[line_polygonlist->polygon[0].v2].x =
 rx;
        line_pointlist->point[line_polygonlist->polygon[0].v2].y =
 -800;
        line_pointlist->point[line_polygonlist->polygon[0].v2].z =
 rz;
        line_pointlist->point[line_polygonlist->polygon[0].v3].x =
 rx-800;
        line_pointlist->point[line_polygonlist->polygon[0].v3].y =
 -400;
        line_pointlist->point[line_polygonlist->polygon[0].v3].z =
 rz;
        hlp = ground_polygonlist->polygon[hl].palet;
        if ( lb )
          ground_polygonlist->polygon[hl].palet = &std_cyan;
      }
  }
        }
        if ( time%2 == 0 ) {
          SetWritePlane( (unsigned short *)0xC00000 );
          minmaxt = minmax1;
        } else {
          SetWritePlane( (unsigned short *)0xC00200 );
          minmaxt = minmax2;
        }

        parameter.x = 0; parameter.y = 50; parameter.z = z;
        TranslateAll( &parameter, work, ground_pointlist, minmaxt
);
        DisplayPolygonList( ground_polygonlist, work, minmaxt );
        minmaxt = AdjustMinMax( minmaxt );

        parameter.x = 0; parameter.y = 50; parameter.z = z;
        TranslateAll( &parameter, work, line_pointlist, minmaxt );
        DisplayPolygonList( line_polygonlist, work, minmaxt );
        minmaxt = AdjustMinMax( minmaxt );

        if ( time%2 == 0 ) {
          HOME( 0, 0, 0 );
          if ( time > 0 ) {
            SetClearPlane( (unsigned short *)0xC00200 );
            ClearBox( minmax2 );
          }
        } else {
          HOME( 0, 256, 0 );
          SetClearPlane( (unsigned short *)0xC00000 );
          ClearBox( minmax1 );
        }
        time++;
      }

    SUPER( sp );

    B_CURON();
    CRTMOD( 16 );

    free( work );

    free( line_polygonlist );
    free( line_pointlist );
    free( ground_polygonlist );
    free( ground_pointlist );

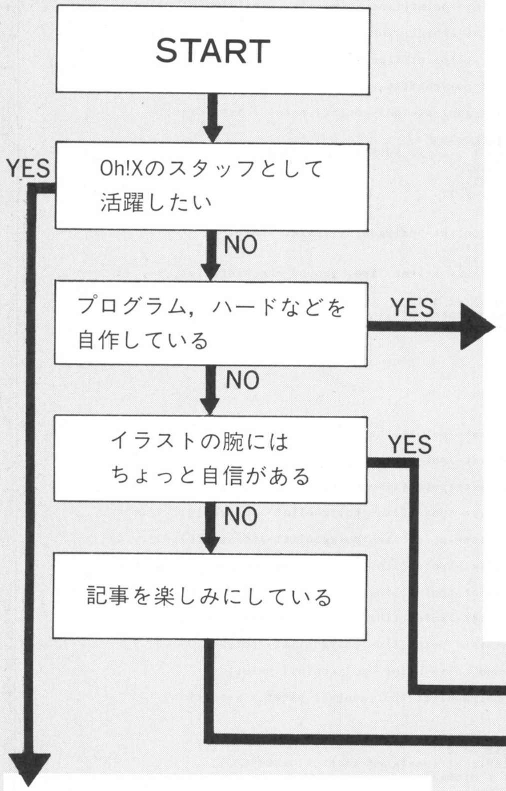
    free( minmax1 );
    free( minmax2 );

    KFLUSHIO( 0xFF );

    return 0;
  }
```

WE WANT YOU!

Oh!Xは，読者の皆さん１人ひとりの力が作り上げていく雑誌です。あなたも誌面作りに協力してくれませんか？



投稿大募集

Oh!Xでは読者の皆さんによる投稿作品を常時募集しています。

未発表の作品であれば，グラフィック，音楽，システムプログラム，ツール，ゲーム，ハードウェアなどジャンルを問いません。機種についても特に限定はしませんが，雑誌の性格上扱いにくい場合もあります。

誌面に載りきらない大きなアプリケーションなどはディスクメディアを使って配布することが考えられます。その形態のひとつはご存じ付録ディスク，そしてもうひとつは別冊形式によるものです（発売中の「Z-MUSICシステムver.2.0」に続き，今後もいくつかのOh!X BOOKSシリーズが予定されています）。

また，「こんなものを作ってみました」といったものでもかまいません。気軽に作品を送ってみませんか。

投稿募集要項

1）　お送りいただくプログラムには，住所，氏名，年齢，職業，連絡先電話番号，機種名，使用言語，動作に必要な周辺機器，パソコン歴などを明記のうえ，封書の宛先の最後には「Oh!X LIVE」「全機種共通システム」「投稿ゲームプログラム」など，プログラムの内容を明確にご記入ください。

2）　投稿されるプログラムには詳しい内容を記入した原稿を同封してください。ディスクの中にドキュメントファイルの形式でのみ記述している方がいますが，郵送時の事故などでメディアが破壊されることもありますので，必ず文書を添えるようにしてください。変数表，メモリマップ，参考文献などの情報があればなお結構です。また，掲載に際しては，プログラムやデータ原稿に対して加筆修正をさせていただくことがあります。

3）　お送りいただくプログラムは事故防止のため最低２回はセーブしておいてください。基本的に原稿などの返送はいたしませんので，あらかじめご了承ください。

4）　ハード製作関係の投稿については，最初は内容のわかる原稿のみお送りいただければ結構です。その後，当方で製作物が必要だと判断した場合には改めてご連絡いたします。

5）　作品の採用については，掲載号が決定した時点で当方より連絡いたします。特にツールやハード関係などの作品は特集内容などを考慮したうえで採用決定されますので，結果を連絡するまで時間がかかる場合があります。

6）　投稿いただいたプログラムにバグなどが発見された場合は，新しいプログラムの入ったメディアと一緒に文書にてご連絡ください。

7）　掲載されたプログラムに対しては当社規定の原稿料をお支払いいたします。また，投稿されたプログラムの著作権などはすべて制作者に保留されますが，いわゆる「フリーソフトなどとしてネットにアップする」ことなどを希望される場合には，必ず事前に編集部までご連絡ください。なお，一般的モラルとして，他誌との二重投稿，または他誌に掲載されたプログラムの移植などは固くお断りいたします。

その他，不明な点は編集部までお問い合わせください。

Oh!X編集部　☎03(5642)8122

協力スタッフ募集

Oh!Xでは誌面作りに参加していただく協力スタッフを募集しています。

スタッフとして活動する熱意があり，東京近郊にお住まいの方でソフトバンクに来社可能な方。時間的束縛は特にありませんが，ある程度時間に余裕がある方に限ります。基本的に学生を対象にしていますが，時間的余裕と余力が十分にあれば社会人も可とします。ただし，18歳未満の学生および浪人生の方については採用予定はありません。

応募要項ですが，ライター希望の方はOh!X誌面１ページ分相当（2500字程度）の自由論文に自己紹介文を添えて「Oh!Xスタッフ希望」係までお送りください。

また，文章力には自信がないけどプログラムなら……という方でも技術スタッフとして参加していただく場合があります。こちらを希望の方は，自由論文の代わりにこれまでに制作した自作プログラムをその解説などと一緒に応募してください。

書類選考後，採用者の方にはこちらからご連絡いたします。

すべての読者へのお願い

いまはまだ何もできないけれど，いつかは……と思っているアナタにも，いますぐできるいちばん重要なことがあります。アンケートハガキへのご協力です。Oh!Xの誌面の方向性は，このアンケートで寄せられた読者のご意見をもとに決定されています。

皆さんからの熱いメッセージをお待ちしています。

そして，宛先

〒103　東京都中央区日本橋浜町3-42-3
ソフトバンク株式会社
Oh!X編集部　○○○○係

イラスト投稿の規定

サイズはハガキ大（A6判）からB5判くらいまでを目安としますが，取り扱いの手間や現実的な問題としてハガキ大を一応の標準とします。いずれにせよ，掲載時にはかなり縮小されることを考慮して描いてください。

一応の推奨形式は以下のとおりです。

1）　ハガキ大のケント紙で郵送

ハガキでも結構ですが，たまに裏面にも消し印が押される場合があります。

2）　黒一色（薄ズミ不可）

墨汁は汚れの原因になることがあります。製図用インクがおすすめです。原稿は縮小されますのでスクリーントーンの80，90番台（レトラセットの場合）や色の濃すぎるものなどについては再現は保証されません。また，残念ながら，カラー原稿はごくたまにしか掲載されません。

内容に関して特に規制はありませんが，季節ものについては，掲載が予想される時期を考慮して早めに送ったほうが有利になることがあります（年賀状は例外）。

皆さんの力作をお待ちしております。

EPA2 補講(その3)

プロジェクトチームDōGA
かまた ゆたか

先月は「森山効果」改め「アニエフェ」の分析の途中で終わってしまいました。今回は，その続きについて実例を豊富に取り入れながら解説し，その応用例も紹介していきます。

はじめに

DōGAでは，毎年冬になると鍋をします。先日は，いわゆる「闇鍋」に挑戦しました。とはいっても，無制限に入れると食べられない鍋になるのはわかっているので，「水炊き，すき焼き，おでんなど，種類は問わないが，鍋の具のみ」という紳士協定を結んで実行されました。

暗闇の中，次々と持ち寄った具が投入されます。そして，お玉で2杯ずつ自分の皿に入れてから，電気をつけました。

「なっ，なんだ? このドロッとしたやつ」

「あっ，それチーズです」

「チーズが鍋の具か!」

「チーズフォンデュ」

「……」

そのとき，乱入してきたSが，チョコレートクッキーを入れ，その時点で，紳士協定は破棄され，鍋は崩壊の道へと歩みます。さつまいも，シーチキン，トマト，リンゴ，キウイなどが投入され，カレーのルーやら麺つゆなどが入れられて，もう鍋なんだか，スープなんだか……。でも，ちゃんと全部食べましたけどね。しかし，さすがに残った汁で雑炊をする奴はいなかったなぁ。

* * *

これまでのあらすじ

西暦2162年。銀河連邦警察官Y.Kamataは，連続する宇宙船の謎の失踪事件を追っていた。目撃者の証言によると，何もない宇宙空間に，突如，稲妻のような放電現象が発生し，宇宙船は，炸裂しながら光に飲み込まれていった(Graphic Gallery参照)という。何らかの新兵器の実験が行われたと考えたKamataは，さっそく，悪の巣窟，惑星ドーガに向かった。

惑星ドーガでは，あやしげな宗教団体ドーガ教が惑星を支配していた。このドーガ教のCGA研究所長森山博士が「EPA2」と呼ばれる最終兵器を開発しており，その効果は「森山効果」あるいは「アニエフェ」というコードネームで呼ばれていた。Kamataは，CGA研究所に潜入し，入手した1枚のMOをもとに，「アニエフェ」の分析を開始したのである。

さて前回は，アニエフェの原理などを解説し，それでは実際に森山氏に実演していただこうというところでプツッと終わってしまいました。ですから，今月号からいきなり読んでも話がわからないと思います。念のため，コラムに「これまでのあらすじ」を用意しましたが，これを読んでも，全然わかりません。ご面倒ですが，ぜひ，先月号を取り出してもう一度読み直してください。

* * *

はい，準備はいいですか? それでは，先月号の続きを再開します。写真Aを見ながら読んでくださいね。

実演，これが爆発だ!

森山：EPA2を立ち上げて，アニメートモードにして，12フレーム目からですね。ふむふむ，11フレーム目は，この位置にビームが貫通しているんですね。このビームの色は?

かまた：あっ，でもビームの色と爆発の色は変えたんですけど。

森：いや，12フレーム目にも，ビームの残留光を入れようと思って。

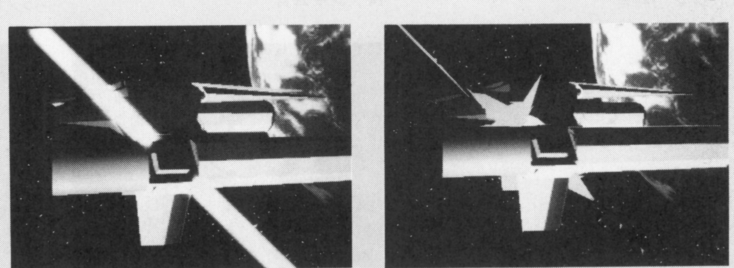
か：なるほど，RGB＝8，25，12 です。

森：残留光は，線で描くのではなく，11フレーム目のビームがあった位置に，点在するように描きます。そして，宇宙船に着弾したあたりに，炸裂光をビームの色で描きます。

か：すべて「Poly」を使ってますね。

森：12フレーム目は，これで終わり。

写真A



11フレーム目　　12フレーム目

か：えっ，これだけ? 結構少ないんですね。

森：着弾してすぐ大爆発っていうパターンもありますが，この爆発は20フレームもあるんですから，最初はブスブスとくすぶっていて，ほかの誘爆が起こり始めてから大爆発，でいいでしょう。

か：おっと，そうこういっているうちに，どんどんフレームが進んでいく(13・14)。この間，すべて「Poly」。すべて，ビームの色ですね。おっと，この15フレーム目は3Dの爆発球にまとわりつくように描いていますね。

森：まぁ，別にすべての場合がこうだ，ってわけじゃないですけど。

か：「Poly」で描くときも，かなり面積の大きいところと，極めて小さいところの差をつけていますけど，これは意識してやっているわけですか。

森：そんなたいしたもんじゃないけど，確かにそれはいえますね。いま16フレーム目ですか。それじゃ一度ここまでアニメーションさせてみましょう。

か：なるほど，やはり確認しながら描いていくわけですね。……いかがですか?

森：まぁ，続けてやりましょか。

か：おっ，17フレーム目で，いきなり「Box」しかも「Fill」。なっ何をするんです。機長! やめてください。

森：えらく古いギャグですね。

か：すんまへん。それはいいけど，いきなり真っ白にしちゃって，いいんですか。

森：いままでブスブスと，弱い効果を与えてといて，ここでいきなり派手にするわけですよ。誘爆が始まる直前に。

か：なるほど。

森：そして，18フレーム目，同様に，「BoxFill」で，ペン先を「Water」にして，薄く白をかぶせます。

か：爆発の余韻を残すわけですね。

森：そして，17フレーム目で黒く穴を開けたあたりに，逆にビーム光を入れます。

か：ちょうど白黒反転するような感じですね。まさに，画面が点滅しますね。

森：ここで，新しい色を設け，稲妻を描きます。まず，「Poly」で描いてから，「Free」で細い線をまとわりつかせます。

か：へぇ，私は結構ピッピッと，いい加減に線を入れましたが，細い線はゆっくり，丁寧に描くんですね。

森：19フレーム目は稲妻だけ。その描き方は同じです。

か：そういえば，周辺に小さな火花がありますね。これは，割と同じ位置に描いていますね。

森：ええ，全部のフレームに描き込んでいるわけではないですが，ゆっくりと遠ざかるように表現するように注意しているつもりです。

か：稲妻や閃光のように一瞬しか入れないものもあれば，ゆっくり動いていくようなものもある。面白い。

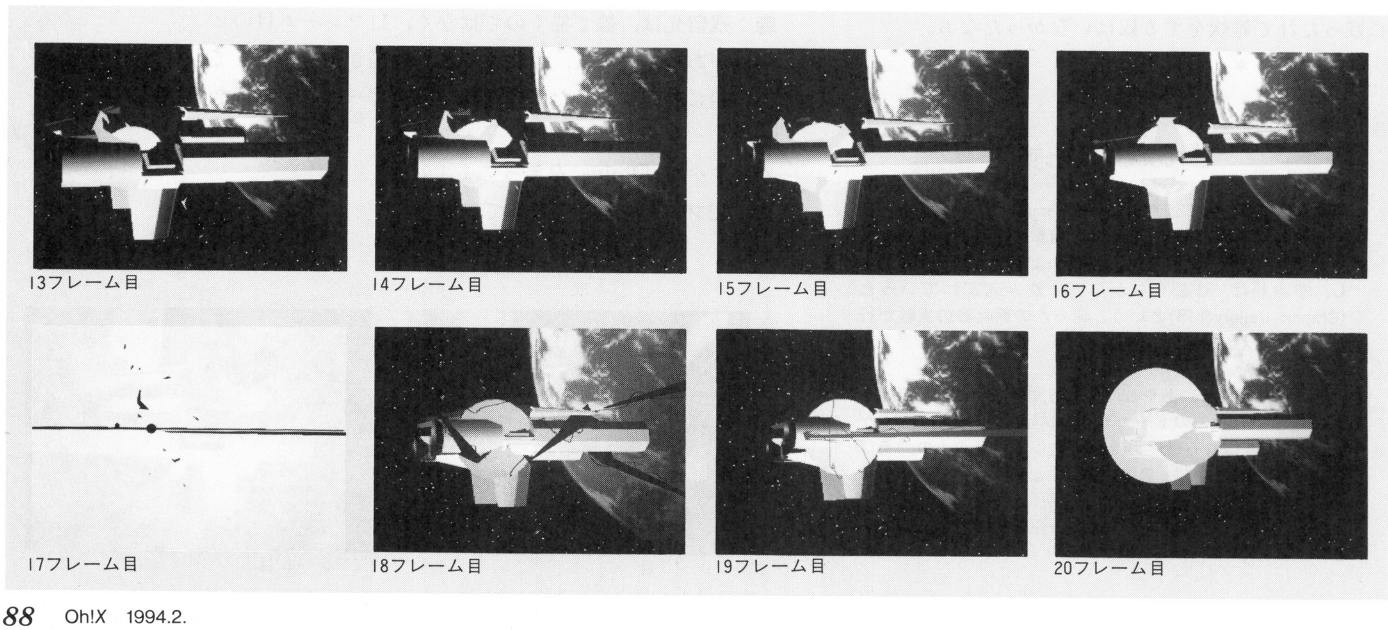
森：20フレーム目では，2発目の爆発が大きくなり始めているので，こいつの閃光を入れてみましょう。まず，「サークルFill」の「Water」ペンで，2発目の爆発を中心に円を描きます。このままだと，3Dの爆発球の色が異なってしまいますので，「Color Change」で元の色に戻しておきます。21フレーム目は，20フレーム目の閃光の円と同心円を大きく，薄く，描きます。22フレーム目でもさらに大きく，さらに薄く描きます。

か：そろそろ，本格的に稲妻が出てきましたね。あれっ? 23フレーム目ですけど，これは細い線だけですか。

森：太い稲妻から，細い稲妻へと変化させてみました。ついでに，次の24フレーム目では，「Poly」も使いましょう。

か：でも，稲妻ひとつとってみても，実にさまざまな描き方をしていますね。直線的なものもあるし，曲線ばか

写真A



13フレーム目　14フレーム目　15フレーム目　16フレーム目
17フレーム目　18フレーム目　19フレーム目　20フレーム目

りのものもあるし，複数の線を絡ませているのや，１本だけ違う方向に伸びるものもある。放射状に伸びるかと思えば，稲妻から稲妻へ橋渡しするようなものもある。
森：ポイントはやはり変化をつけるということですね。さあ，最後の大爆発を前に，26フレーム目で，また画面を真っ白にする閃光を入れましょう。ついでに「サークル」で，ちょっと中心のずれた円を２つ描いて，その間を塗りつぶして，細くなったところを白で消します。
か：あれっ，27フレーム目はとばすのですか?
森：あとで，ちょっと稲妻を加えますが，先に28フレーム目を描きましょう。
か：これも，画面真っ白ですね。ちゃんと点滅させているわけだ。
森：ちょっとフレームが足りなくなっちゃいましたね。これでは，タメが足りないかな。あと３フレーム欲しかった。27フレーム目に手を加えて，はい出来上がり。
か：こうしてみると，実にたくさんのテクニックが使われていますね。
森：今回は，強引に描き上げましたが，実際に自分の作品では，１回で思いどおりになることはないですよ。何回も試行錯誤します。
か：今回の爆発の出来はいかがですか。
森：まぁ，派手ですね。でも，ほかの人が作ったカットにアニエフェを加えるのは難しいですねぇ。自分が原画

▶CGAコンテスト事務局より◀

さぁ，CGAコンテストを間近に控えて，だんだん盛り上がってきた。各担当者も動きだしており，いろいろと連絡事項があるからチェックしておいてほしい。

■総合プロデューサーの紹介

第６回アマチュアCGAコンテストの総合プロデューサーはマリオ古本だ。また，昨年の総合プロデューサーの遊び人松井がアシスタントプロデューサーを務める。マリオ古本がいうには，今回のコンテストは単なる上映会の枠を破り，アマチュアCGA界の総合イベントとして，より「お祭」化するそうだ。つまり，作品上映以外に来場者参加型のイベントを用意するらしいが，詳しいことは当日まで秘密とのこと。

■コンテストビデオデザイナーの紹介

毎年配布している，CGAコンテストのビデオ。年々凝ったものになってきているが，今回のトータルデザイナーとして，「手紙」や「X68000イメージビデオ」のプーロ北川さんが登用された。トータルデザイナーって何をするのかというと，実はよくわからない。北川さんが「コンテストのビデオも，単なる作品の寄せ集めではなく，全体としてひとつの映像作品になっているような演出をデザインするべきだ」といったので，「じゃあ，やってよ」ということになったのだ。どういったものになるかはまだ企画中だが，なんか楽しみだな。ちなみに，今回のCGAコンテストのオープニングアニメーションは，KMC(京大マイコンクラブ)が制作する。どのようなものに仕上がるか，こっちも楽しみだ。

■コンテストビデオのプレ申し込み中止のお知らせ

毎年，たいへんご好評をいただき，今年も楽しみにしている方が多いと思われるCGAコンテストのビデオ配布ですが，今回は諸般の事情によりプレ申し込みは受けつけません。その代わり，Oh!X４月号に申し込み用紙をつける予定です。ということで，今年はプレ申し込みをしないように注意してください。

■現地スタッフ

東京地区上映会の，現地スタッフにご応募くださった皆さん，ご協力に感謝します。もうそろそろ第１回ミーティングのお知らせが届いていると思います。私もおじゃまします。「申し込もうと思ってたけど，忘れてた」という方は，いまならまだ間に合いますので，お手紙ください。

〒533　大阪市東淀川区淡路5-17-2-102号
プロジェクトチームDōGA
「CGAコンテスト事務局」まで

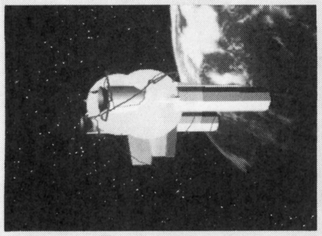
■コンテスト日程と会場

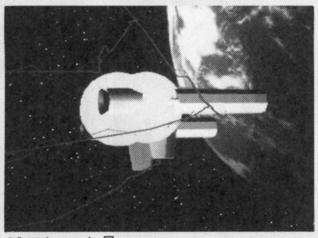
最後に，コンテストの場所と日時をもう一度，確認します。詳しくは来月号に掲載しますが，とにかく，この日にほかの予定を入れないようにちゃんと調整しておいてくださいね。

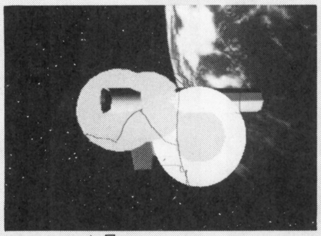
・東京地区：３月６日(日)
千代田区永田町　国会議事堂横
三宅坂ホール
・大阪地区：４月２日(土)
摂津市千里丘
摂津市民文化ホール

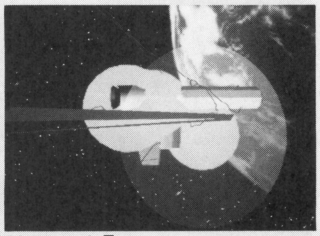
ところで，三宅坂ホールの下見に行った遊び人松井が会場の外からの写真をたくさん撮っていると，国会議事堂を警備している警察官に呼び止められたそうだ。当日，会場に向かうときは，エアガン，サバイバーショット，角棒などを持ってたら，きっと，ただではすまないぞ。そんな人たちが並んでたら，変な集会と思われて中止になったりして……。冗談じゃないぞ。

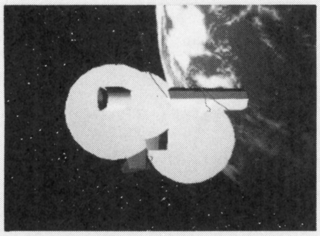
写真A

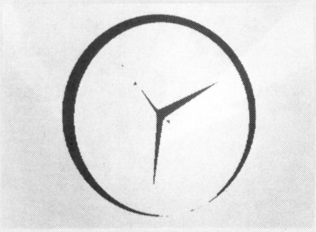

21フレーム目

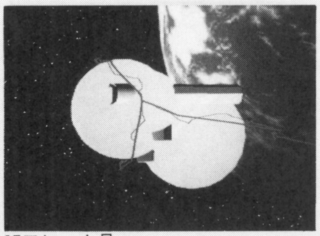

22フレーム目

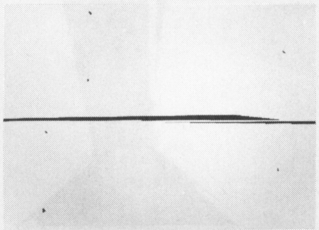

23フレーム目


24フレーム目


25フレーム目


26フレーム目


27フレーム目


28フレーム目

を作るときは，何フレーム目にどうなって，何フレーム目からどうなるというのもちゃんと考えて作るしね。

レーザー砲発射

森：アニエフェは，爆発だけに使えるテクニックではありませんが，基本は同じです。あとはいろいろ試してみてください。

か：まぁまぁ，そういわずにもう1つぐらい例を挙げて解説してくださいな。

森：そうですか。じゃあ，レーザー砲の発射シーンでも作ってみましょう。

か：まずは，簡単な描き方をお願いします。

森：簡単な発射シーンは，2フレームでできます。1フレーム目に，砲口に丸い光を描きます。遠慮せず，大きめに描いてください。2フレーム目は，砲口からほとばしるビームを「Poly」で描きます。画面のパースに注意してください。レーザーなんだから砲口の反対側はフレームアウトしていること。最も無難な例がこれです。結構それらしく見えるし，2フレームで終わるので楽です。

か：確かに楽ですね。では，もう少しアレンジするとどうなりますか。

森：まず，3Dの原画をいじりましょうか。発射の2フレームにだけ点光源を使ってみましょう。このとき，光源の強さはフレームごとに変えましょう。そして，この2フレームだけは平行光源による通常照明をなくして，全体的には暗くしてやります。レーザー光の明るさを強調するわけですね。

か：前回の例にも挙げましたね。

森：ほかには，反動もつけましょう。つまり，このカットの何フレーム目にレーザーを発射するというのを考えておいて，その次のフレームでレーザー光と反対の方向に動かしてやるわけです。写真Bの例では，ちょっとおおげさに反動をつけてみました。

か：なるほど，反動をつけるとレーザーのパワーが強調されますね。EPA2で描き加える工夫はないですか?。

森：点光源を使ったことで，レーザー光によって物体が照らされている部分ができたと思います。そのなかで特に明るくなっている部分を，EPA2で上描きして，そこを光らせるのも面白い効果が出ます。これは，宮崎アニメでもよく使われる由緒正しい手法です。

それから，前回も，宇宙空間など周りに何もない場合は，光源を工夫しても光を受ける物体がないので効果が出ないという話をしましたが，そういうときは，宇宙空間自体を無理やり白っぽくさせるという手法もあります。

か：さっきも思ったのですが，なぜ，何もない宇宙空間が光るのですか?

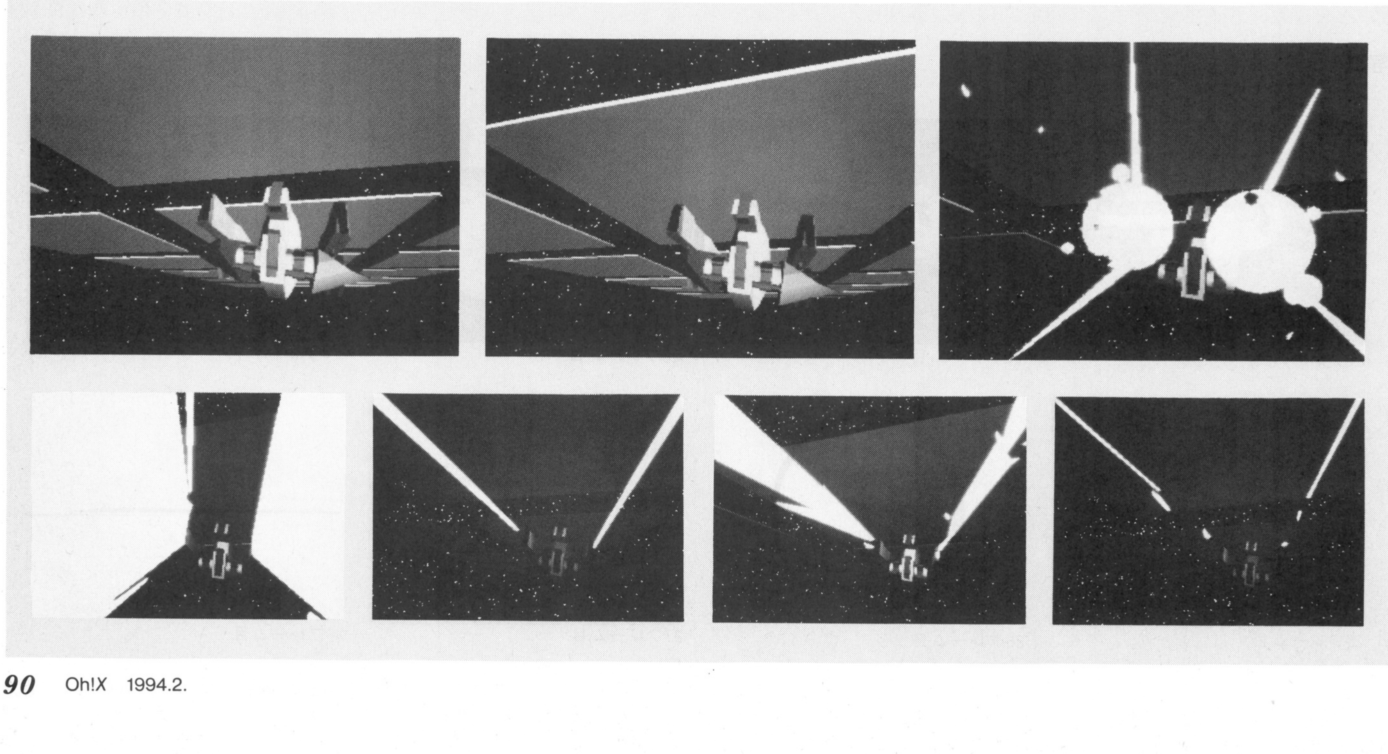
森：物理法則の介入する余地などありません。視覚的に最も衝撃的な方法を選択すべきです。

また，1フレーム目を「丸」じゃなくて，もっとかっこいい形にアレンジするとか，3フレームぐらい使って丸をふくらませるとか，2フレーム目の「線」を数フレームに引き伸ばし，その間だんだん細くなっていくように描く。あるいは丸から線へなめらかにつなぐ……。

か：いろいろあるわけですね。写真Bのレーザーも工夫してますね。

森：レーザー光線の太さがころころ変わるわけはないのですが，これも演出ですね。太い線，細い線を交互にして点滅させるなどは，基本ですよ。皆さんも，いろいろ試してみてください。

写真B　反動をつけてレーザーのパワーを強調する



DōGA

応用へのアイデア

以上，森山さんに，アニエフェの与え方をひと通り解説していただきましたが，これですべてというわけではありません。まだまだいろいろな与え方があります。工夫次第で，皆さん独自の表現も見つかるでしょう。

私もいくつか変わった与え方を実験してみましたので紹介します。複数併用するなり，オリジナルのアニエフェの参考にしていただければ幸いです。

1)　ギザギザ爆発(写真C)

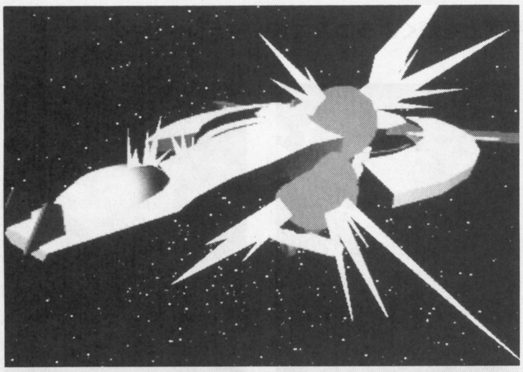
爆発は，球だけとは限りません。写真のようなギザギザの爆発も力強い感じが出ます。爆発球と同様に，3Dのポリゴンで作るという方法もあるでしょうが，やってみてもうまくいきませんでした。というのは，静止画なら問題はないのでしょうが，動画の場合，このギザギザ爆発は，激しく動いていないと感じが出ないからです。EPA2で描き加えるときも，アニメートモードを活かして，前のフレームと大きく異なるようにするか，爆発球が出現する寸前に数フレームだけ入れるといった使い方が効果的でしょう。

2)　2色使用(写真D)

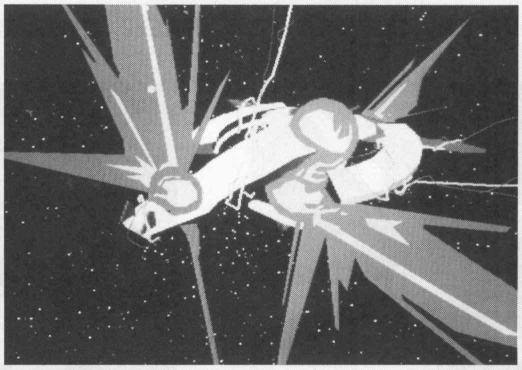
透過光とする色を2色にするという手法です。3色以上でも実験してみましたが，あまり効果は上がらず，手間ばかり増えるので意味がないでしょう。

2色にするケースとしては，3つ考えられます。まず，稲妻を2色に描くケースでは，「Poly」で描く太い稲妻と，最小のペンで描く細い稲妻を別の色にすると効果があります。2つ目のケースは爆発球です。ポリゴンで作った球はやや暗めにしておいて，EPA2でその球の中心

写真C　ギザギザ爆発



写真D　2色使用



夫婦でQ&A

うさ子：遅ればせながら，明けましておめでとうございます。年賀状ありがとうございます。すべての方に返信できなくてすみません。

村松さん(「D」の作者)：**「CGAマガジン編集部御中」**

うさ子：これはいったい何でしょうか?

ゆたか：正直者にしか見えないディスクなんとちゃう?

うさ子：だったら，私には見えるはずでしょ。

ゆたか：ということで，村松さん。CGAマガジンへの投稿はディスクケースだけではなく，ちゃんとディスクも入れて送ってください。

Oさん(葛飾区)：**12月号のような，うだつの上がらないお役人は，般若の面つけて，ぶっとばしちゃえ。現地スタッフの募集については，頭をあまり使わない仕事なら希望します。**

ゆたか：現地スタッフへのご応募ありがとうございます。さっそく，CGAコンテスト担当者から連絡させていただきます。また近々，DōGA東京支部を作るという計画もありますので，その節はご参加ください。なお，お役人の件につきましては，仕返しが怖いですので，ご遠慮させていただきます。

うさ子：……。

ゆたか：どうしたん?

うさ子：……「うだつ」って何?

ゆたか：……さぁ?

Oさん(練馬区)：**この度，セルロイドアニメータをやめることを決意しました。2年間の専門学校での勉強の末，ようやく動画の仕事の下っぱを始めたのですが，どうしても，次の問題には2に○をつけてしまうからです。**

問：どちらを取るか?

1．セルアニメーション

2．コンピュータ

だから決めました。とはいえ，アマチュアでもいいから今後も映像を作りたいのです。

うさ子：がんばってくださいネ(細川ふみえ風に)。

ゆたか：最近，えらい細川ふみえに執着してんねんな。

うさ子：だって，胸が大きいんですもん。

ゆたか：……?　よぅわからんけど。

Aさん(山形)：**私はX68030ユーザーで，030に対応しているCGAシステムを入手したいと思っています。その場合はどうすればよいのでしょうか。**

うさ子：030対応といっても，たいした変更はしていませんが，一応最新バージョンを入手されたいときは，遠慮なく当チームへ申し出てください。申し込み用紙をお送りします。1992年7月号の申し込み用紙を流用しても問題ないでしょう。通信欄に，何を希望されるのか記入してください。ディスク代，発送費などは適当で結構です。

Yさん(東京)：**DōGAでは，CGAマガジンやコンテストの際など，著作権には注意していると思っていたのですが，なぜCGAマガジンに「PHALANX」が入っていたのですか?**

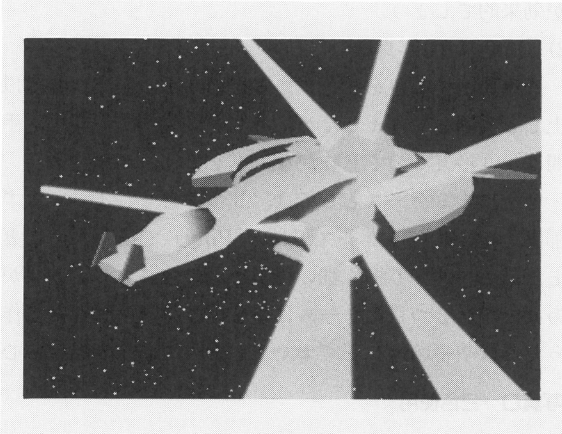
うさ子：それはですね，ちゃんとズームの許可を取っているからです。

ゆたか：ズームの皆さん，ありがとうございます。

うさ子：非営利目的に使用するのでしたら，敵メカなどもかまわないそうなので，どなたか，「PHALANX」の完全3D化に挑戦しませんか?

付近にもう少し明るい球を描き加えるという方法です。そして3つ目はギザギザ爆発で，爆発球同様，内側に少し明るい(光らせたとき白っぽくなる)色でギザギザを描くという方法です。

写真E　放射光



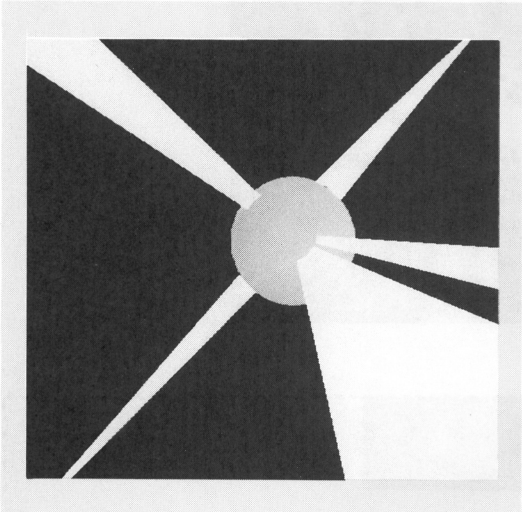
2色使うと，色を変えるという手間が思いのほか大変なのですが，効果はありますので試してください。なお，色はまず最初に「PALET」に保存しておくと作業が楽になります。

3)　放射光(写真E)

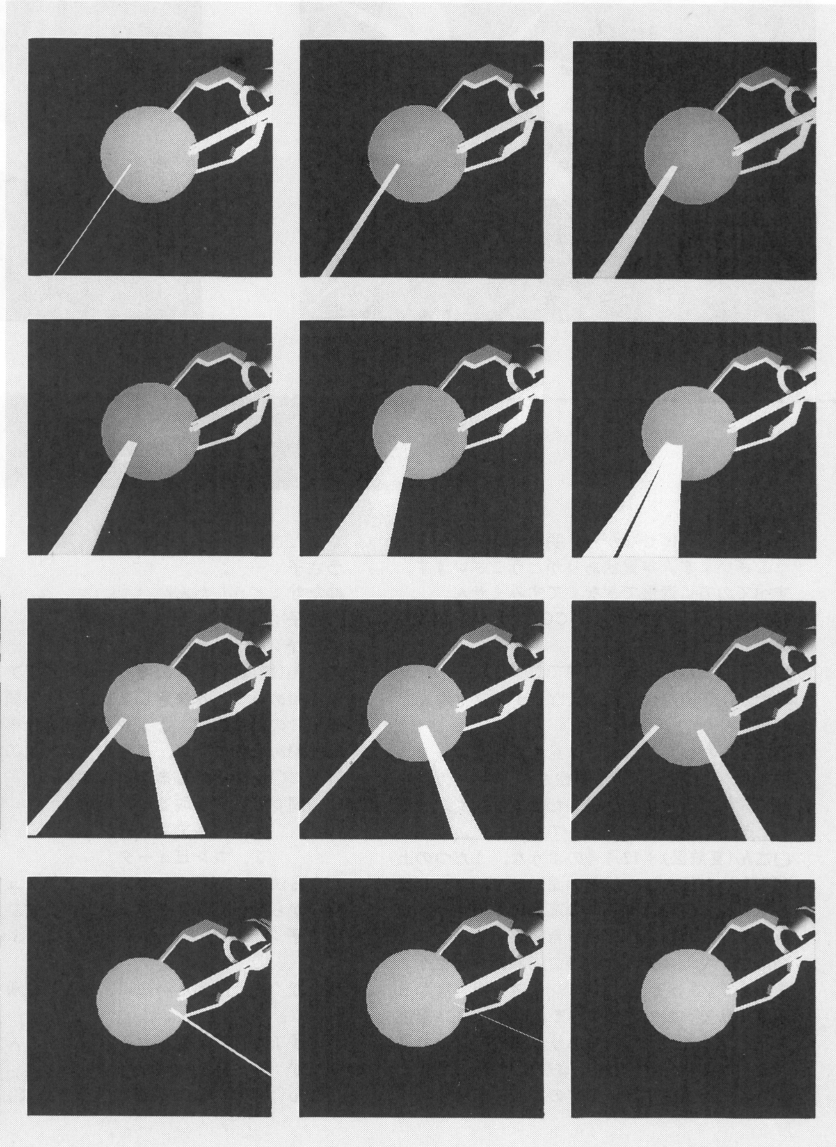
ギザギザ爆発をさらに発展させた形といえるかもしれません。爆発球を中心に，閃光を「Poly」で画面の端まで放射状に描きます。この場合，はじめからかなり白っぽい色を設定し，透過光の機能を使わないという手法も考えられます。光っているという感じは多少劣りますが，エッジが鋭くなるというメリットがあります。また，「Water」ペン(Opacity 6)を使っても面白いかもしれません。

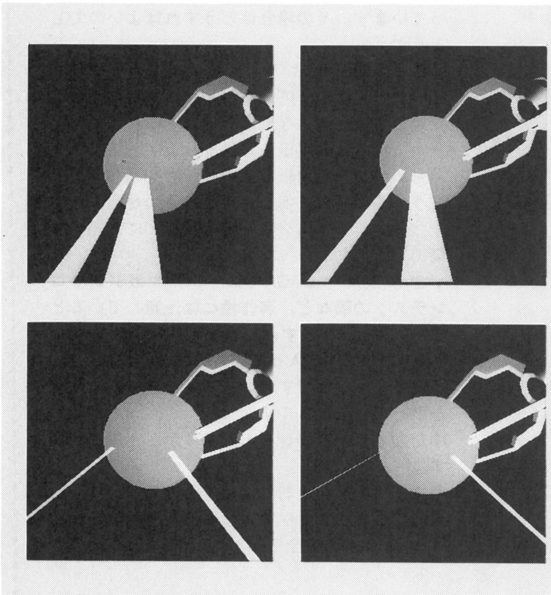
この放射光，簡単なように見えますが，いろいろとコツがあります。まず，写真E-1を見るとよくわかりますが，爆発球の手前側から出ている放射光は手前へ伸びてくる光ですから，先へ行くほど太くします。逆に，爆発

写真E-1　3次元的広がりを考慮して放射光を描く



写真E-2　放射光は閃光が同一平面上を動くようにアニメーションさせる





球の裏側から発せられる光は，先を少し細くします。ちゃんと3次元的な広がりを計算に入れているわけです。

次に動かし方の問題があります。この放射光，ほかの手法と異なり，前後のフレームと関係なくランダムに与えても，うまく感じが出ません。閃光が同一平面上をスーッと動いているようにアニメーションしてやります。写真E-2をご覧ください。ここでは，最初細かった光が太い閃光になり，それが2つに分かれて広がっていき，細くなって消えていくというふうに動かしてみました。閃光が太いときはゆっくりと，細くなるにつれて速く移動させます。1つの閃光が現れ，消えていくのは，7～17フレームぐらいでしょう。

ほかにもいろいろな動かし方があるでしょうし，少し複雑になりますが，ポリゴンで処理する方法も面白いでしょう。

4)　同心円(写真F)

放射状があるなら，同心円状もあるだろうということで描いてみましたが，あまりうまくできませんでした。第一，結構面倒くさいです。うまくやればカッコはいいかもしれませんが，これをアニメーションさせるのはかなり手間がかかるでしょう。私は途中で挫折しました。根性がある方は挑戦してみてください。

5)　オーラ(写真G)

剣にオーラが発生しているような雰囲気に挑戦してみましたが……失敗しました。やはり，こういったものはある程度時間をかけて修業を積まないといけませんね。

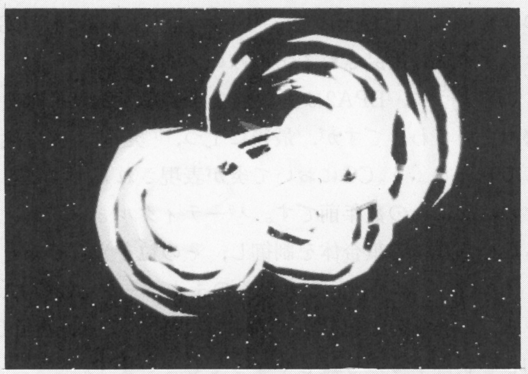
描き方は，物体をやや暗くレンダリングしておいて，物体のアウトラインをなぞるように，凹凸をつけながら「Poly」で作画します。これは，爆発する寸前とか，静電気がたまっている様子を描くときに使えるのではないでしょうか。オーラを描くときは，「Free」で曲線を主体に描いたほうがよいかもしれません。

マッピング爆発球

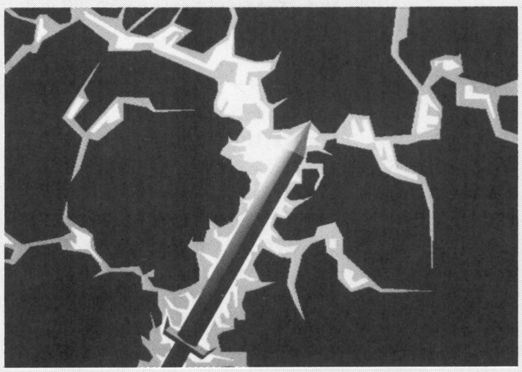
EPA2の解説からは脱線しますが，マッピングを使って爆発球を工夫してみました(写真H)。単なる輝く球体ではなく，煙も混じった感じになります。マッピングというと手間がかかりそうですが，TAMEN.XはUV座標をつけることができるので，たいした手間にはなりません。

問題となるのは張りつける画像ですが，これもいろいろ工夫のしがいがあります。煙だからといって，雲のようなものを丁寧にたくさん描いていっても，あまりそれらしくなりませんでした。むしろ，フラクタルなどの細かいランダムっぽい模様のほうが適当でしょう。

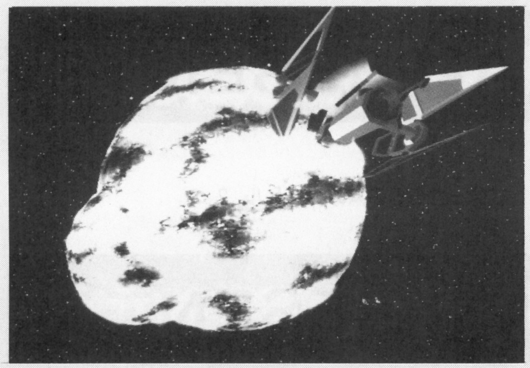
写真F　同心円



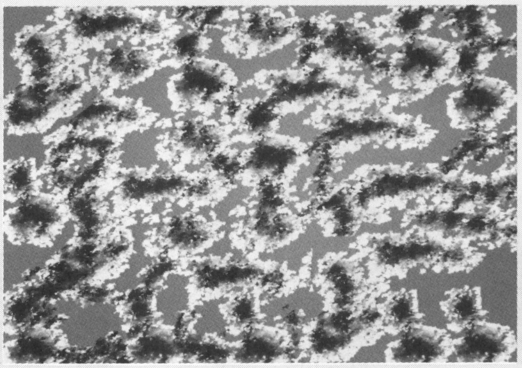
写真G　オーラ



写真H　マッピング爆発球



写真H-1　マッピングする画像を作成



私は，MATIERのにじみペンを使って写真H-1のような画像を作ってみました。雲の周辺部は白っぽく，真ん中は厚みがあるので黒っぽくしています。地のベタッとしているところがあとで光らせる部分になるわけです。

アニエフェ研究

このように，アニエフェのテクニックや応用は，書いていて尽きるものではありません。その場その場に応じて描き方が異なるケースも多々あります。どのような場合にどのような描き方をするのか？　具体例は，漫画やアニメを見れば，たくさん紹介されています。

森山さんがおっしゃるには，「アニエフェは，日本のセルアニメにおいて一部のアニメーターたちによって構築された技術の模倣にすぎません。このアニメーターの技術は，世界に誇っていいものです。しかし，マイナーな技術なので，それに関する本が出版されたということは聞きません」。

ということだそうなので，皆さんもアニメ作品などを見て，1コマずつチェックして，いろんなアニエフェを研究しましょう。森山さんにお勧めの作品を紹介していただきましたので参考にしてください(コラム)。

EPA2のさらなる応用

以上，いろいろなアニエフェについて解説してきましたが，アニエフェは，なにも爆発や稲妻だけにしか使えないテクニックではありません。工夫次第でほかにもいろいろ応用できます。

たとえば，第5回CGAコンテストのオープニング，エンディングアニメのなかには，スペキュラーをアニエフェで与えるというカットがあります(写真I)。また，写真J, Kは，DōGAのタイトルをEPA2を利用して作ったものです。どうやって作ったかは，ホームズのように推理してください。

さらに，未確認の情報ですが，海中の光の表現にEPA2を利用したという話も聞いています。何でも，滑らかな曲面で作られたイルカを動かし，そこに現れる不規則なマッハバンドの1色を光らせるそうです。すると，海中のゆらゆらとした光になる……本当でしょうか？　CGAコンテストに出品されると面白いのですが。

このように，EPA2は実にさまざまな表現の可能性を秘めているわけですが，最後に1つ，「炎」の表現に挑戦してみましょう。CGにおいて炎が表現されるようになったのは，ほんの数年前です。パーティクルというランダムな小さな粒の集合体を制御し，その粒を光らせるというアルゴリズムです。これをそのままパーソナルCGAでやると，多くの粒を計算するにはCPUパワーが不足し，動きを制御するのも難しくなってしまいます。

しかし，ランダムな粒の集合体は，3D－2D変換した

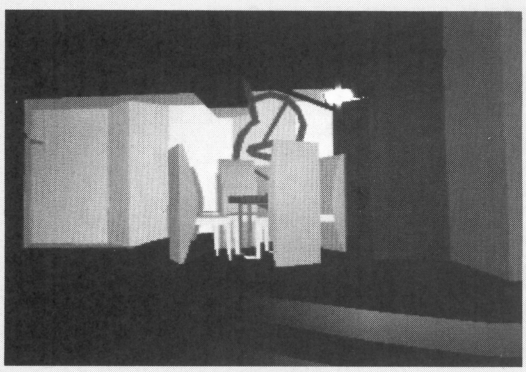
森山さんお勧めのアニメ作品

●さよなら銀河鉄道999
　ラスト30分は圧巻!
●BIRTH(バース)
　全編これでもか!　だけど，尻すぼみになっていくのが悲しい。
●風の谷のナウシカ
　部分的にいい。どこがいいかは自分で探そう。
●天空の城ラピュタ
　上に同じ。
●SF新世紀レンズマン
　CGよりセルが頑張っている作品だ。
●幻魔大戦
　後半の火龍など，見るべきところは多い。
●クラッシャージョウ（劇場版）
　この作品リストにおいては異質だが，ビギナーはこのあたりを参考にするのが安全かもしれない。
●ヴィナス戦記
　上に同じ。全編見せ場という，とっても疲れる劇場作品。
●巨人ゴーグ
　上に同じ。これはテレビシリーズだが，全編を通じ完璧なまでに作画品位が安定していた。
●うる星やつら　オンリーユー（最初の劇場版）
　個性的な戦闘シーン，無茶なスピード感。描写がいちいち大胆である。
●迷宮物語
　オムニバス作品。どれもなかなかいいが2本目（だったかな?）の「走る男」のスピード感が極上。押井守の「迷宮物件」ではない(あれはあれで必見の作品ではあるが)。
●ダロス
　これも「オンリーユー」と同じ理由による。ただ，あれよりはアニメーターが自制しているようでリアリティはちょっとあるかな。

＊　＊　＊

なんだか趣味に走っているなあ，古い作品ばっか。最近の作品はあまり知らないし。たまに見ても，最近はリアリティに走っているのか「うまい」作画は結構あるが，「大胆な」作画にはほとんどお目にかかれない。つまらん。

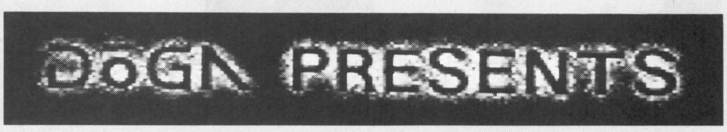
写真I　スペキュラーをアニエフェで与える



写真J　EPA2を利用して作ったDōGAのタイトル 1



写真K　EPA2を利用して作ったDōGAのタイトル 2



あともランダムな点の集合体であるという点に着目すると，非常に簡単に扱うことができます。2D上のランダムな点の集合体，これはペイントソフトのエアブラシのことです。これをEPA2で光らせると，見事な炎が出来上がります。

ペイントソフトで炎の原画を描くのが難しそうですが，写真Lをご覧ください。実に適当に，サラッと描いただけでも，十分炎に見えます。エアブラシは「Free」の「Pen」の「Shape」を「Brush」に切り替えて使います。色は暗めで(23, 10, 5)ぐらいがよいでしょう。太めの筆でさっと描いて，炎の先のあたりはやや細い筆で簡単に整えます。

原画ができたら，「Random Fire」で「Str23」ぐらい強く光らせます。そして，「Edit」の「Soft Focus」の「Normal」を2回かけます。

このように静止画はいたって簡単なのですが，ちゃんとアニメーションさせるとなると，ちょっと練習が必要です。写真Mをご覧ください。出来の悪いサンプルで申し訳ないのですが，こんな感じで8フレームを繰り返せば燃えているように見えます。

炎もいろいろありますが，この例のような焚火程度の大きさなら8～10フレーム程度必要です。マッチの炎ならもっと少なくてよいでしょう。

まず，1フレーム目を適当に描きます。2フレーム目は，アニメートモードで透かしながら前のフレームの筆の塊の少し上に移動して，やや小さく描きます。そして，いちばん下の根元に空きができますので，新しい塊を加えます。炎が上へ上へと動いている感じを出すわけです。

1つの筆の塊は，発生から消滅まで，4～5フレームで描きます。1フレーム目で根元に発生し，2フレーム目で大きくなり，3～4フレーム目で小さくなって，5フレーム目は火の粉になるわけです。塊の形など，きれいにトレースする必要は全然ありません。炎全体の形もかなり変化してかまいません。とにかく光らせてぼやけさせれば炎に見えますから，あまり1枚1枚にこだわらなくてよいでしょう。

さらにそれっぽく見せるためには，3Dの画像にも工夫が必要です。まず当然ですが，炎の位置に赤い点光源を配置します。炎は大きさがありますから，黄色っぽい光源と赤っぽい光源を上下に2つ配置するのもよいかもしれません。そしてここがポイントなのですが，その光源の明るさ(色)を乱数で変動させてやるのです。ロウソクやガス灯ならともかく，焚火の場合，明るさはかなり変化します。ただ，光源が明るいフレームは，炎も大きく描くといった配慮はまったく無用です。そんなことは，全然わかりません。

おわりに

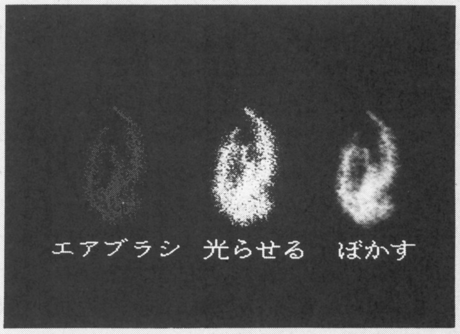
いや～，今回のEPA2補講は大変でした。まずいろいろな表現方法を自分自身で試行錯誤しなくちゃいけないし，サンプル画像をたくさん作らないといけません。また，サンプル画像を作る過程で，さらにまた面白そうなアイデアが浮かんでしまいます。きりがありません。

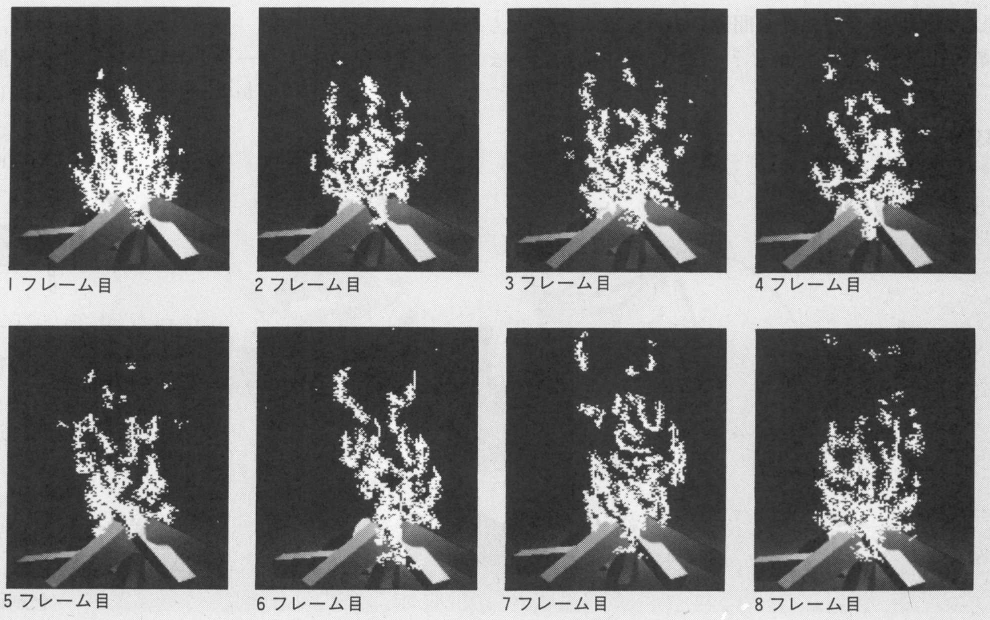
3回にわたって，かなり詳しく解説を行ってきましたが，これを隅から隅まで読んで暗記しても，アニエフェができるとはいえません。何といっても，実際にやってみないとうまくならないでしょう。何度かトライしてみて，どうもうまくいかないようでしたら，もう一度この連載を読み直してください。きっとヒントが見つかるはずです。

さて，次回は外部の方に原稿を依頼しています。その方は，私の妹の家庭教師をされていたのです。世の中狭いですね。話が見えない？　それは次回のお楽しみということです。

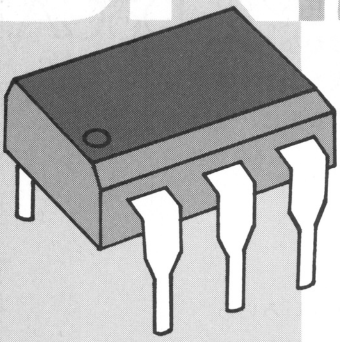
それでは，また。

写真L　炎の原画を描く





ONE CHIP IC



ワンチップIC工作入門（第3回）

マイクロムービングマシンを操る

Takao Katsuhiko
高尾 克彦

久しぶりに登場の「ワンチップIC工作入門」。今回はエフェクタ関係からちょっと離れて，全長30mmのマイクロロボットの制御を行ってみます。工作は簡単ですので，ぜひチャレンジしてください。

“マイクロマシン”。それは，科学，医療などの分野で注目を集め，いま急速に発展している超小型マシンの総称。バンダイでは世界に先駆け，この技術をホビーに導入。そして誕生したのが，M³(エムスリー)＝マイクロムービングマシン！　マイクロコイルとスーパーマグネットの磁石を応用し，2足歩行を実現した超小型ロボット。全高約30mmのアクションホビーの登場！

（パッケージより）

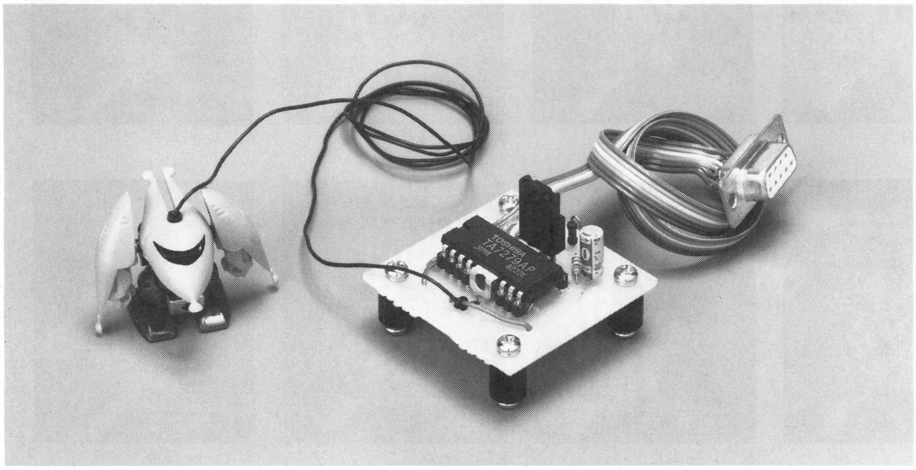
だそうで，写真1が，ちょっと前まで小学生の間ではやっていたM³ロボットです。

写真1では，このロボットのすごさがわかりづらいと思いますが，とにかく近くのおもちゃ屋さんへ行って実物を見てください。本当に全長30mmのちっちゃなロボットが，ちょこちょこちょこ，と2足歩行(!!)し，しかも2チャンネルのコントローラによって，ストップ←→前進，右折←→左折，そして，歩行モードの選択（ノーマル，スキップ）までできてしまうのです。

ともかく，これはすごいことです。

今春，めでたく，機械工学の学士号を修得した私がいうのだから間違いありません(あれ，理工学だったかな？　まぁ，どちらでもいいや)。

写真1　マイクロマシンM³



今回は，X68000を用いてこのロボットを動かしてみます。

とにかくバラす

パッケージにも「対象年齢10歳以上」と書いてあるとおり，さすがに1時間も遊んでいると飽きるので，X68000への接続を考え始めます。

もうひとり，このM³を買った友人を見つけて，「対戦」という遊び方も考えられますが（競走，相撲，応用技で9人探してバスケットボール，21人探してラグビー），とにかくX68000への接続を考えます。

リモートコントローラに後退ボタンがないことからも，適当にロボットの足をばたつかせていれば，前進することがわかります。さらに，方向レバーを右に入れたり左に入れたりしていると，ロボットが旋回することがわかってきます。

スキップモードというのも，実は足を交互に動かすのではなく，一緒に両足を同じ方向に動かして実現しているのだということもわかります。

また，リモートコントローラとロボットをつなぐケーブル（というかコード）のコネクタを見ると，4本の信号がロボットへと渡されていることがわかります（むき出しの導線が1本と，赤，緑，青が各1本)。

ロボットは精密に作られていて，おいそれとバラすわけにはいきませんが，それでも股の隙間から覗き込んでみると，股間にコイルが2種類あって，その力で脚の制御を行っているようです。

ということは，ひとつのコイルの入力が＋極と－極の2つ。それが2倍で信号線が4本と考えるのが自然です。この4本が，直接ロボットのコイルにつながっているというのは間違いなさそうです。適当に電圧をかけてみた結果，

むき出しの導線，赤色の導線　：右足
緑色の導線，青色の導線　　　：左足

という接続になっているようです。

また，送られてくる信号をオシロスコープで観察した結果，±3Vの矩形波が送られていることがわかりました。このパルス幅を制御することによって，ロボットの動作スピードを制御しているようです。

つまり，リモートコントローラからの命令は，すべて両足をばたつかせるタイミングを制御することによって，実現されているわけです。

以上のことからロボットを動かすには，X68000から2つのコイルを制御できるようにすればよいということになります。

電源について

まず，ロボットのコイルを駆動させるための電源を確保しなければいけません。このコイルに，3Vの電圧を正方向にかけたり，逆方向にかけたりして，ロボットの足をバタバタさせます。

ジョイスティックポートからは，0V，5Vの信号が出ていますので，電源，右足用の信号，左足用の信号の計3本でM³ロボットを制御できるようになります。この際，電

圧の正確な値はどうでもいいとして，変動しない2.5Vの電圧が確保できなくてはなりません。しかし，この変動しない2.5Vの電圧というのがくせ者で，たとえば，

右足を前に：制御用信号(5V)→電源
右足を後ろに：制御用信号(0V)←電源

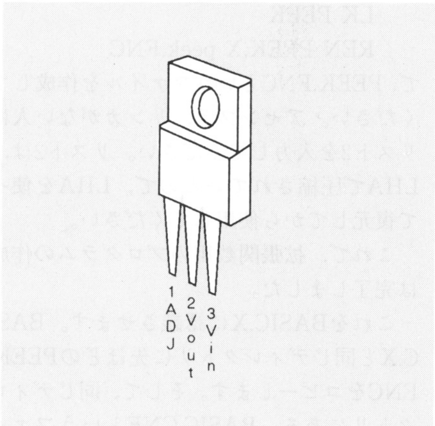
のように，電流を吸い出すときにも，吐き出すときにも，ともに2.5Vとなっていなければならないものです。

結論からいってしまうと，このような電源を確保するのは，無理ではないとしても，かなり面倒な作業です。トランジスタやOPアンプを使って，定電圧を作るのはそれほど難しいことではありません。しかし，電流の逆流が生じたときにはトランジスタやOPアンプの保護を行わなければなりませんし，逆向きのトランジスタなども用意しなければならなくなってしまいます。

そんなわけで，絶対変化しない2.5Vというのは，作り出せたら素敵だけれど，なかなかどうして現実は難しいので，今回は見送ることにします（なりました)。

そして，電流を吐き出すときだけに電圧を保証するLM317というICを使って，定電圧を確保することにします。

図1　LM317のピン配置図



LM317

定電圧の作成といえば，三端子レギュレータと呼ばれるICを使うのが定石となっています。この三端子レギュレータというのは，アバウトな電圧を入力端子に加えることによって，出力端子からきっちりと希望の定電圧が得られるという，便利なICです。三端子レギュレータには，これらの入力，出力端子のほかに，0Vに接地するためのGND端子があって，計３本の端子が出ているという，名前が正直に体を表しています。

ふつう，三端子レギュレータというと，ほとんど無条件に，75××とか76××などという型番の（定電圧）三端子レギュレータと呼ばれるものを使うことになっています。

前述の3Vを得るために三端子レギュレータのカタログをパラパラーっとめくると……ない，のです。きっかり，3Vを出せるやつが。12V用，7V用，5V用ときて，3V用はなぜかありません。メーカーのほうも1Vおきに製品を用意してもいられないのでしょうか。ま，ぷーたれていてもしょうがないので，さらに別の手段を考えます。

定電圧三端子レギュレータに希望するものがないのなら，可変電圧三端子レギュレータと呼ばれるものの中から，適当なものを探します。可変電圧三端子レギュレータにもいろいろなものがあるのですが，おそらく最もポピュラーで，値段も安く入手しやすい，ということで今回はLM317を採用しました。LM317は，内部でなにをやっているかといえば，図２を見てもわかるとおり，巨大なエミッタフォロワのようです。

データブックによると図３のような回路の場合，出力は，

$$\text{Vout} = 1.25\times\left(1+\frac{\text{R2}}{\text{R1}}\right)$$

と，なるようですから，R1＝240Ω，R2＝300Ωとして，3Vの電圧を得ています。

TA7279A

次にモーターやコイルを制御するには，

1)　ON，OFFの選択
2)　ONの場合の電流の向きも制御する

以上の２つのほかに，場合によっては，

3)　電流の大きさ

まで制御してやらなければならないときもあるかもしれませんが（モーターの速度制御など)，今回は2)までです。

1)はスイッチング回路といって，図４のようにトランジスタ１個で簡単に実現できます。

問題なのは2)の制御です。理屈はともかく，これを実現するには図５のような回路が必要になってきてしまいます。モーターの制御を考えれば，必ず出てくる回路なのに，実現するにはこんなに面倒くさい，というわけで，いかにも１チップLSIで実現されていそうな回路です。

おそらく，いろいろな種類のLSIがあるんだろうな，と思いつつカタログをめくっていると意外と少ないのです。

ON/OFF制御用にはいっぱい出ているのですが，正・逆転の制御までとなると，かなり絞られてしまいます。しかも（M³ロボットの右足と左足の）２チャンネル分をいっぺんに制御できるようなもの，というのはさらに絞られてきて，そうこうしているうちにTA7279AというLSIに落ち着きました（図６，７，表１，２)。

図６を見ると，図５のような回路が出力部に組み込まれてるので，とりあえず安心します。しかし，よく見るとダイオードが

図2　LM317の内部等価回路

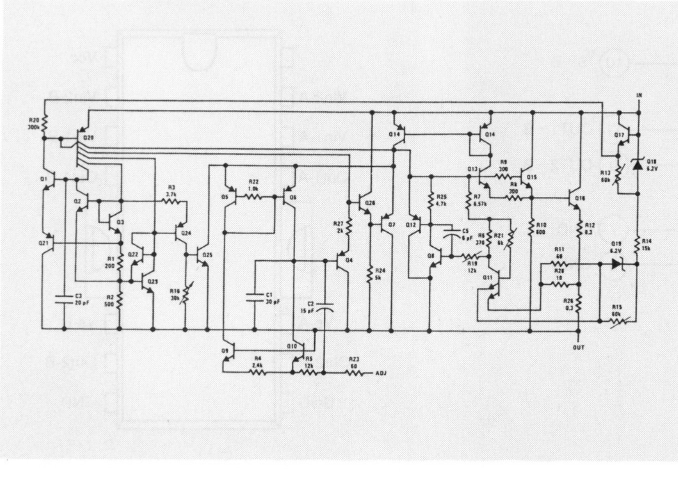
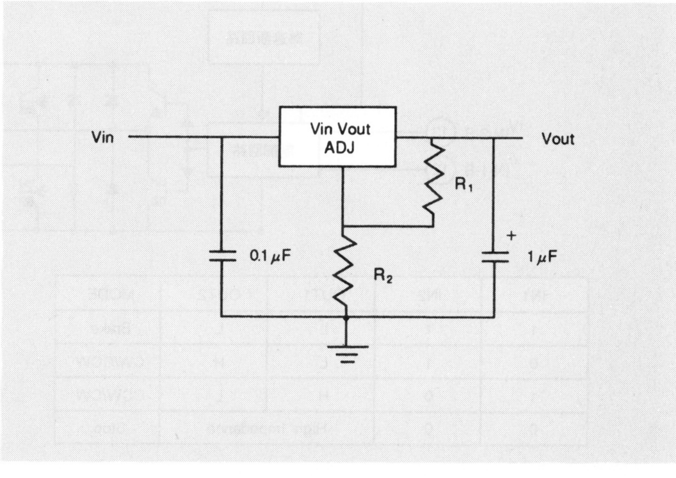


図3　LM317基本回路



それぞれのチャンネルに各4本，計8本挿入されています（これは，モーターのコイルからの逆起電力の流入を防ぐもの）。

逆起電力というのは，電流を急激に変化させた場合，それを妨げる方向に生じるというコイル特有の電力です。この電圧の大きさは，変化の急激具合に比例して大きくなります。今回の回路では，瞬時にモーターの電流を切ったり入れたりするわけですから，理屈のうえでは，この逆起電力というのは，ものすごい大きさになってしまうわけです（Δt=0ですからね)。

受験生の皆さん，ファラデーの法則，

$$V = -L\frac{dI}{dt}$$

は大丈夫ですね。

それはともかく，このTA7279Aでは，そのような逆起電力が逆流してこないように，図6のダイオードが入っているのです。そのほかにも，フライホイールダイオードの役割を果たしていて，これがあるとモーターが滑らかに回転をするとかいうご利益があるのですが，今回はモーターの制御ではなく，コイルなのであまり関係がありません。とにかく，TA7279Aはここまで，モーター(コイル)の制御に徹したLSIなんだな，ということで安心して製作に進みます。

インタフェイスを作る

回路図を図8に，その写真を写真2に示します。今回は，TA7279Aに必要な電源&信号をつなぐ，というだけで残りはほとんど直結していますから，いつにもまして単純な回路構成です。基板には，サンハヤトのICB-97を2つに分断したものを使用しました。

図4 ON/OFFのみのモーター制御回路

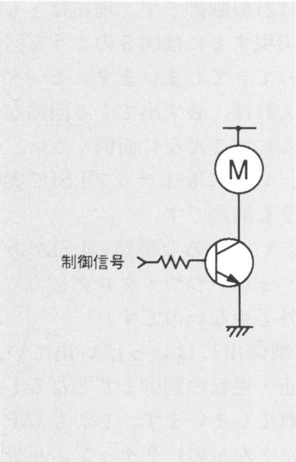


図5 正・逆転制御のモーター制御回路

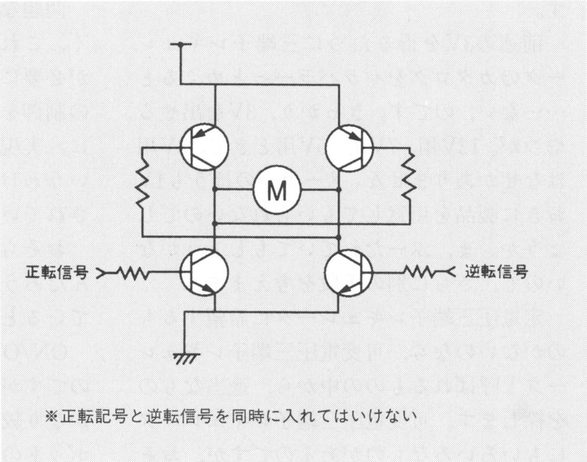
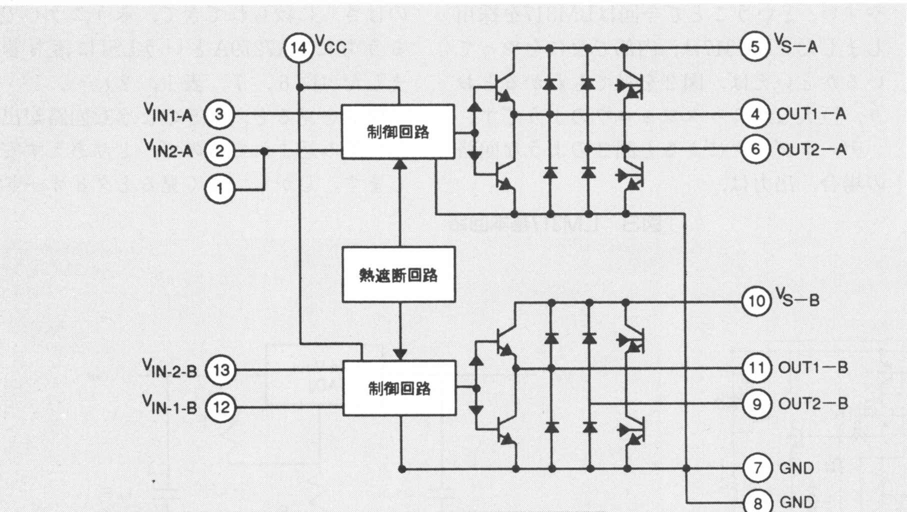


図6 TA7279Aのブロック図



IN1	IN2	OUT1	OUT2	MODE
1	1	L	L	Brake
0	1	L	H	CW/CCW
1	0	H	L	CCW/CW
0	0	High Impedance		Stop

拡張関数

今回の製作でも，X68000から信号を引き出すのに，ジョイスティックポートを使用しました。

普通，ジョイスティックポートというのは，データ入力に使われるポートなので，出力用に使うには，X68000内部の8255というLSIの設定を入力→出力へと変更してやらなければなりません。そのためには，アドレス&HE9A006に，データ&H83を書き込まなければならないのですが，X-BASICからこのような動作はできません（なぜこのようになるかは，桒野氏の「Inside X68000」を参照)。できないことはどうするかというと，いつものように，あきらめるか自力でなんとかするしかありません。自力でなんとかするためには，リスト1のプログラムを入力します。エディタを立ち上げて，リスト1を入力したら，PEEK.Sというファイル名でセーブし，

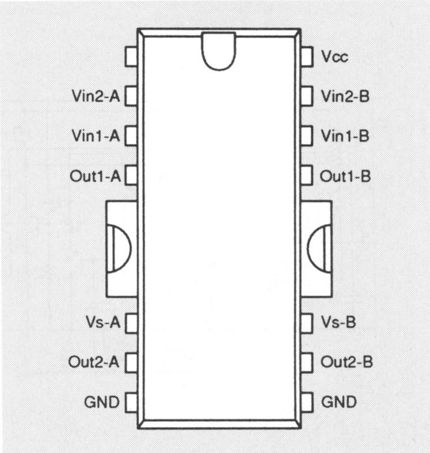
AS PEEK
LK PEEK
REN PEEK.X peek.FNC

で，PEEK.FNCというファイルを作成してください。アセンブラ，リンカがない人は，リスト2を入力してください。リスト2は，LHAで圧縮されているので，LHAを使って復元してから使用してください。

これで，拡張関数用のプログラムの作成は完了しました。

これをBASIC.Xに認識させます。BASIC.Xと同じディレクトリに先ほどのPEEK.FNCをコピーします。そして，同じディレクトリにある，BASIC.CNFというファイ

図7 TA7279Aのピン配置図



ルに，

FUNC ＝ PEEK

という1行を追加します。

これで任意のアドレスを直接アクセスできるようになりました。以上の作業で拡張された関数は以下の2つです。

・poke（式1，式2）

式1で示されるアドレスに，式2の値を代入する。戻り値はなし。

・peek（式1）

式1で示されるアドレスに代入されているデータを戻り値として返す。

回路のチェック

回路が組み上がったら，いよいよ動作チェックを行います。

例によって簡単な回路ですので，日頃の行いと腕に自信のある方は，すっ飛ばしていきなりロボットを動かしてみてもかまいません。

そうでない人は，回路図の左のほうからチェックしていきます。

TA7279Aはまだ挿入せずに，回路をX68000のジョイスティックポート1（前面にあるほう）に接続します。X68000の電源を入れてみて，通常どおりの動作が行われることを確認してください。ここで，コケてしまった人は，もう理由がハッキリしています。電源のショートです。もう一度，回路を確かめましょう。

次に，LM317で無事に3Vが作成されているかを確めます。LM317のVout端子（真中の奴）にテスターを当て，その電圧を確認してください。

厳密な調整ではないので，テスターを持っていない人は買わなくてもかまいません。多少不便ですが，豆電球を代わりに使います。まず，豆電球に3Vの電圧を加えたときの明るさを目で覚えます。3Vというのは，普通の乾電池を2本直列につないだときの電圧です。この，豆電球の明るさというのは，加えた電圧にある程度比例しますから，電球の明るさからおおよその電圧がわかるわけです。

図8 回路図

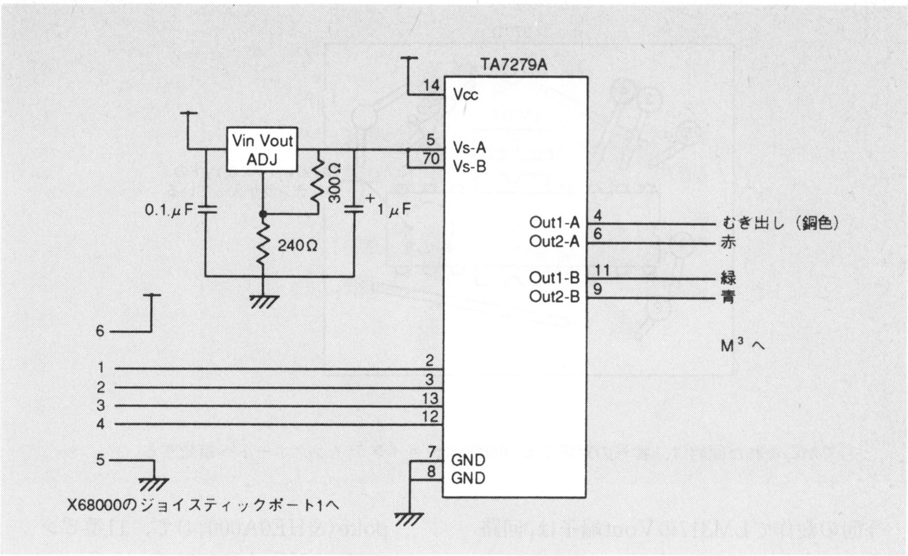


表1 TA7279Aの最大定格

項目		記号	定格	単位
ロジック側電源電圧	AP	$V_{CC(MAX)}$	25	V
	P		20	
出力側電源電圧	AP	$V_{S(MAX)}$	25	V
	P		18	
出力電流	ピーク	$I_{O(PEAK)}$	3.0	A
	AVE.	$I_{O(AVE)}$	1.0	
許容損失		P_D	2.3	W
動作温度		Topr	−30〜75	℃
保存温度		Tstg	−55〜150	℃

表3 部品表

IC	TA7279A	500円	1個
	LM317	100円	1個
抵抗	240Ω	@10円	1個
	300Ω	@10円	1個
コンデンサ	0.1μF	@10円	1個
電解コンデンサ	1μF	@10円	1個
基板	ICB-97	100円	1枚
ケーブル		100円	少々
コネクタ		300円	1個

表2 TA7279Aの電気特性

項目		記号	測定回路	測定条件	最小	標準	最大	単位
電源電流		I_{CC1}	1	V_{CC}＝18V 出力 OFF STOP	14	28	41	mA
		I_{CC2}	1	V_{CC}＝18V 出力 OFF CW／CCW	10	29	38	
		I_{CC3}	1	V_{CC}＝18V 出力 OFF BRAKE	8	20	35	
入力電圧	1 (High)	$V_{N(H)}$	−	T_j＝25℃, 2, 3 12, 13ピン	3.0	−	Vcc	V
	2 (Low)	$V_{N(L)}$	−	T_j＝25℃, 2, 3, 12, 13ピン	−	−	0.8	
入力電流		I_{IN}	2	シンク V_{IN}＝3V	−	3	10	μA
飽和電圧	上	V_{SATU-1}	3	I_O＝0.1A, V_{CC}＝V_S＝18V	−	−	1.1	V
	下	V_{SATL-1}	3	I_O＝0.1A, V_{CC}＝V_S＝18V	−	−	1.0	
	上	V_{SATU-2}	3	I_O＝1.0A, V_{CC}＝V_S＝18V	−	1.2	1.5	
	下	V_{SATL-2}	3	I_O＝1.0A, V_{CC}＝V_S＝18V	−	1.05	1.4	
出力トランジスタリーク電流	上	I_{LU}	−	V_S＝25V	−	−	50	μA
	下	I_{LL}	−	V_S＝25V	−	−	50	
ダイオードフォワード電圧	上	V_{FU}	4	I_F＝1A	−	2.5	−	V
	下	V_{FL}	4	I_F＝1A	−	1.3	−	

写真2 回路

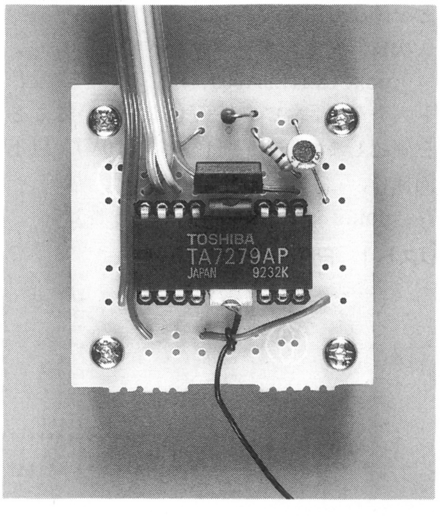
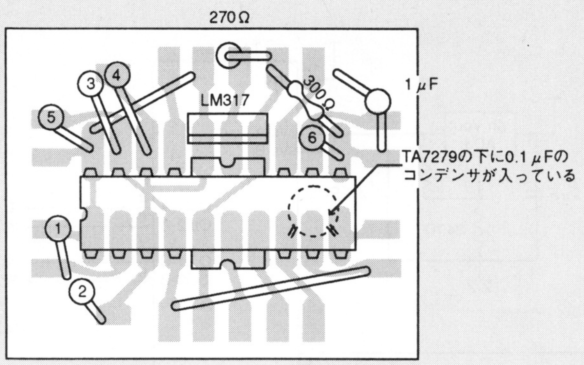


図9　実体配線図（表）



○でかこまれた配線は，番号の対応するX68000のジョイスティックポートへ接続する

今回の製作でLM317のVout端子は，回路がうまく動作していれば3V，していなければ0V，5V，というように，これら以外の電圧を取りようがありませんから，このような簡単なチェックで問題はありません（テスターに関しても，以下同じ）。あ，たいていの豆電球は5Vに接続すると壊れるので気をつけてください。

電源のチェックがすんだら，いよいよコイルの制御部分のチェックです。いったんX68000の電源を抜いてから，TA7279Aをソケットに差し込み，再びX68000の電源を入れます（まだこの状態でもロボットを接続しないほうが無難でしょう）。

Human68kのコマンドラインから，X-BASICを立ち上げます。この際，先ほどの拡張関数も組み込んでおいてください。

ダイレクトコマンドで，

poke(&HE9A006,&H83)

と打ち込み，ジョイスティックポート1の1～4番ピンを出力用にします。この状態で，TA7279Aの出力ピン(4，6，9，11番ピン)にテスターを当てて0Vになっていることを確認してください。

次に，ダイレクトコマンドで，

poke(&HE9A000,1)

と入力し，4番ピンが3Vになっていることを確認します(6，9，11番ピンは0V)。以下同様に，

poke(&HE9A000,2)で，6番ピン
poke(&HE9A000,4)で，11番ピン
peek(&HE9A000,8)で，9番ピン

それぞれのピンが，3Vになっていることを確認してください。

これまでのチェックで特に問題がない人は，回路が正常に動いています。電源は正しく得られているのに，出力がうまくいかないという人は，おそらくX68000からの信号とTA7279Aの入力端子との接続に問題があります。

プログラムについて

先ほどもいったとおり，M^3ロボットは，足を前後にばたつかせるだけで動きます。前後にゆっくり足を動かせば，ゆっくり歩きますし，バタバタと素早く足を動かせば駆け足で前に進みます。

これを，左右の足を互い違いに動かせば普通の動かし方となりますし，一緒に動かせばスキップとなります。

また，左右の足を動かす間隔をずらすと，真っ直ぐに前進せずに，横方向にそれていきます。足を前後に1回動かして進む「1歩」の距離は一定ですから，同じ時間でも，速く足をバタバタさせると，より遠くへ(つまりはより速く）動くわけです。右足をゆっくり，左足を素早く動かせば右側にそれていきますし，逆に，右足を素早く，左足をゆっくりと動かせば，左側にそれていきます。

それをプログラムしたのが，リスト3です。このプログラムをRUNすると，M^3がゆっくりと歩きだします。ここで，8のキーを押すと加速し，2のキーで減速します。左右の旋回は，それぞれ4で右，6で左となっています。

また，Sキーを押すと，スキップを始めます。プログラムの内容は簡単ですので，詳しい説明は省略します。使っている変数は，

rc：右足をバタつかせる間隔
lc：左足をバタつかせる間隔
d：ジョイスティックポートへ出力する値となっています。

最後に

今回はM^3を制御してみましたが，TA7279AというICは，電圧25V（ピーク，平均で18V)，電流3.0A(同，1.0A)というかなり大きな電力を扱えるようになっています。また，LM317は可変型の三端子レギュレータで，1.7～37Vの電圧が自由に取り出せます。つまり，R2の値を適当に変えてやれば，たいていのモーターの制御は行えてしまうわけです。

模型やおもちゃというのは，ほとんどがモーターで動きますから，今回の回路でかなりの数の模型が接続できるようになったわけです。

今回は，M^3に付属のコントローラをX68000のキーボードに置き換えただけでした。しかし，プログラムの組み方さえわかれば，メモリに記憶させておいたとおりの動きを再現させることもできます。さらに，ロボットになんらかのセンサを取りつけて，それをX68000で拾うことができればいろいろと面白いことができそうです。

M^3にセンサを取りつけるのは不可能に近いことですが，たとえば，物音がしたらシンバルを叩き出すチンパンジーのぬいぐるみとかなら簡単に作れそうです。

参考文献

1）最新電力用素子規格表，CQ出版

リスト1　PEEK.S

```
                .nlist
_SUPER          equ     $ff20
int_val         equ     $0002
int_ret         equ     $8001
void_ret        equ     $ffff
arg_typ         equ     0
arg_cnt         equ     4
arg_vec         equ     12
                .list
***********************************
* インフォメーション テーブル *
***********************************
        dc.l    _ret
        dc.l    _ret
        dc.l    _ret
        dc.l    _ret
        dc.l    _ret
        dc.l    _ret
        dc.l    _ret
        dc.l    _ret
        dc.l    _token
        dc.l    _param
        dc.l    _exec
        dc.l    0
```


▶家のテレビ（1982年製）は，部屋が寒いとワイドテレビになる。　武田　誠(19)千葉県

```
        dc.l    0
        dc.l    0
        dc.l    0
        dc.l    0
****************
* プログラム *
****************
        .even
_ret:
        rts
        .even
_super:
        clr.l   -(sp)
        dc.w    _SUPER
        addq.l  #4,sp
        move.l  d0,_sspbuf
        move.l  usp,a0
        move.l  a0,_uspbuf
        rts
        .even
_user:
        move.l  _uspbuf,a0
        move.l  a0,usp
        move.l  _sspbuf,-(sp)
        dc.w    _SUPER
        addq.l  #4,sp
        rts
        .even
*******************
* レジスタリード  *
*                 *
*  regr(addr)     *
*    addr : int   *
*                 *
*******************
_regr:
*
        bsr     _super
        move.l  arg_vec(sp),a0
        clr.l   d0
        move.w  (a0),d0
        move.l  d0,_ret_val
*
        bsr     _user
*
        clr.l   d0
        lea     _ret_arg,a0
        rts
        .even
**********************
*   レジスタライト   *
*                    *
*    regw(addr,val)  *
*       addr : int   *
*       val  : int   *
*                    *
**********************
_regw:
        bsr     _super
*
        move.l  arg_vec(sp),a0
        move.l  arg_vec+10(sp),d0
        and.l   #$ffff,d0
        move.w  d0,(a0)
*
        bsr     _user
*
        clr.l   d0
        rts

*****************
* データ エリア *
*****************
        .even
_token:
        dc.b    'peek',0
        dc.b    'poke',0
        .even
_param:
        dc.l    _regr_param
        dc.l    _regw_param
        .even
_regr_param:
        dc.w    int_val
        dc.w    int_ret
_regw_param:
        dc.w    int_val
        dc.w    int_val
        dc.w    void_ret
        .even
_exec:
        dc.l    _regr
        dc.l    _regw
        .even
_ret_arg:
        dc.w    0
        dc.l    0
_ret_val:
        dc.l    0

        .even
_sspbuf:
        dc.l    0
_uspbuf:
        dc.l    0
        .even

        end
```

リスト2　PEEK.LZH（セーブバイト数　209バイト）

```
0000  21 50 2D 6C 68 35 2D AD  : 81
0008  00 00 00 40 01 00 00 64  : A5
0010  AD FE 14 20 01 08 50 45  : 7D
0018  45 4B 2E 46 4E 43 E1 60  : D6
0020  48 00 00 00 81 52 56 DA  : 4B
0028  35 24 FA E3 21 11 11 25  : 9E
0030  96 E2 C8 C4 8E CA C9 D7  : FC
0038  25 25 C9 C1 29 64 38 20  : B9
0040  D8 36 47 90 93 E5 24 E7  : 68
0048  64 E0 94 90 89 0C 86 42  : C5
0050  42 24 24 22 24 2F 02 A4  : A5
0058  BB FE C9 52 75 2D 27 4C  : E9
0060  8A 36 DC C5 8C 7A CE DE  : 13
0068  53 BC 3F E6 0F C7 51 1D  : 78
0070  84 68 F9 38 95 A3 5B 4F  : FF
0078  93 4C 14 92 9C D3 9A 0E  : 9C
------------------------------------
SUM:  78 A2 EA 83 92 15 AD 1D  C214
```

```
0080  25 DF 44 AC F9 EA EF 2C  : F2
0088  37 DC DD D0 29 5C 7D 4F  : 11
0090  30 B2 86 45 6E A8 DE 51  : F2
0098  F7 43 30 F8 24 78 13 BD  : CE
00A0  EE 76 78 43 F4 95 9D AD  : F2
00A8  89 91 D0 86 25 73 A9 52  : 03
00B0  09 E6 91 4E B6 6D 23 B8  : CC
00B8  19 95 81 81 9A D2 56 52  : C4
00C0  3A 3E FD 57 BD ED C1 5E  : 95
00C8  AD 7C 58 17 49 7B 41 E0  : 7D
00D0  00 00 00 00 00 00 00 00  : 00
00D8  00 00 00 00 00 00 00 00  : 00
00E0  00 00 00 00 00 00 00 00  : 00
00E8  00 00 00 00 00 00 00 00  : 00
00F0  00 00 00 00 00 00 00 00  : 00
00F8  00 00 00 00 00 00 00 00  : 00
------------------------------------
SUM:  03 EC 86 BF 23 15 1E D0  A553
```

リスト3　M3.BAS

```
 10 str a
 20 int d
 30 int turn, speed, rc, lc
 40 cls
 50 locate 0,0: print "右足 左足"
 60 locate 0,2: print "Normal Mode"
 70 skipmode = 0
 80 turn = 0: speed = 5
 90 d = 5
100 poke(&HE9A006,&H83)
110 while(1)
120   a = inkey$(0)
130   if a="2" then speed = speed + 1
140   if a="8" then speed = speed - 1
150   if a="4" then turn = turn + 1
160   if a="6" then turn = turn - 1
170   if a="s" then inkeys()
180   if speed > 10 then speed = 10
190   if speed < 0  then speed = 0
200   if turn  >  5 then turn = 5
210   if turn  < -5 then ruen = -5
220   if(skipmode) then {
230     rt = rt - 1
240     if rt < 0 then rt = speed: d = d xor 15
250     rc = speed: lc = speed
260   } else {
270     rc = speed - turn: if rc < 0 then rc = 0
280     lc = speed + turn: if lc < 0 then lc = 0
290     rt = rt - 1
300     if rt < 0 then rt = rc: d = d xor 3
310     lt = lt - 1
320     if lt < 0 then lt = lc: d = d xor 12
330   }
340   locate 0,1 : print rc, lc
350   poke(&HE9A000,d)
360 endwhile
370 end
380 /*
390 func inkeys()
400   skipmode = skipmode xor 1
410   locate 0,2
420   if skipmode = 1 then {
430     print "Skip Mode  "
440     d = 9
450   } else {
460     print "Normal Mode"
470     d = 5
480     rt = 0: lt = lc / 2
490   }
500 endfunc
```

▶「TG100」を中古で買いました。YAMAHAはいいですね。　　藤沢　吉文(17)山梨県

CREATIVE COMPUTER MUSIC

Creative Computer Music入門(29)

最終回 転調と借用和音

Taki Yasushi 瀧　康史

予定より早めですが，今月で最終回となってしまいました。実践に始まり，理論や原理までを解説したなかからみなさんは何をくみ取ってくださったでしょうか。今後も音楽を楽しむうえでのなんらかの手助けになることを祈りつつ，2年半の連載の幕を閉じさせていただきます。

寒い。

もちろん気温はちょうどよいのがいちばんだけど，暑いよりは寒いほうがずっといい。冷え性ではないからそれほど辛くはないし。もっとも，東京などという軟弱な場所に住んでいるから，そういえるのかもしれないけど(実は静岡からまた引っ越したのです)。北海道なんかに住んでいたら，そうはいえないだろうな，きっと。

寒いっていうのはなかなか情緒があっていいと思う。肌にピリピリした感じが，自分としてはとても好きかもしれない。体温を徐々に奪われていく感じがいいかなーなんて思うこともあるな。なんだか自然のなかに引きずり込まれていくような……。別にマゾヒスティックな快感に浸っているのではなくってね。

そういう感じから，メロディなんかが私はよく浮かんだりする。強引に曲作りしようとして作られた曲って，結構世の中にあふれてるけど(特にポップスなんかで)，こういうときに知らずと浮かんだメロディのほうが，自然で美しい場合が多いんだよね。曲を聴く数が増えれば増えるほど，それは痛切に感じるな。あえていわないけど，これは思い入れなしに仕事で作った曲だなっていうのが見え見えのもあるしね(そういうのは美しくないでしょ?　やっぱり。それとも，美学の違いかなぁ)。

詩的であること。

たぶんこれは，曲を作るうえで重要なことなのだと思う。

ドラマチックな生活というのは，いくらでも周りに広がってるし。人を待つためにコーヒーを飲みながら，ふと時計を見る。そういったありふれた仕草でも，表現によってはドラマチックに着飾ることはできるはずだしね。大切なのはその心理描写なんじゃないかな?

え?　何がいいたいのかって?

ただ単に，私が曲を作るときのシチュエーションを書きたかっただけなんだけど。私の場合，思いついたら手帳に楽譜を書きなぐるかなぁ。だから，電子手帳は使えないんだよね。電子手帳の住所管理はすっごく魅力的だけど，いくらPalmTopみたいなのでも，楽譜はちょっとねぇ。

曲を思いつくときのニュアンス。これは私個人の場合ですけど参考になったでしょうか?　それでは最終回，始めることにしましょうか。

§主旨に関すること

和声学を勉強し始めると，どうもあることを忘れてしまいがちになる傾向があります。それは，和声学が絶対ではないという前提です。

和声理論が絶対ではないと頭ではわかっていても，これがあなたの感性のブレーキになってしまうようでは笑い話にもなりません。多少の「いわゆる和声学的な」ミスにこだわってしまうあまり，曲が作りにくくなってしまったら，元も子もないでしょう?

和声にこだわった曲を作ろうとする場合ならばそれでもよいのですが，私がこれまで連載してきた内容は，むしろ目安としての知識に近いのです。だから当然，理論に当てはまらない曲というのもあるわけです。そして，理論に当てはまらなくても，よい曲というのは世の中いっぱいあります。

以前，アーティストがプロとしてやっていけるのは，和声理論を熟知しているからではなくて，そのアーティスト個人の感性が素晴らしいからだといいました。もちろん，音楽で収入を得ている人が和声理論をまったく知らないわけはないですけれど。売れているアーティストと売れていないアーティストの差は，そのアーティスト個人の感性に同調する人がどれだけいるか，ということだけだという考え方もできなくはないですからね。

崇高な音楽だとか，稚拙な音楽であるかとかいうようには一概に決めることができないところが，芸術のよいところです。「あれは音が外れているから駄目だ」とかいって，そういう曲をかたくなに拒否している人がいますが，外れている音って具体的に説明できますか?　また，外れている音って絶対に美しくないものでしょうか?　意図的なことなのかもしれませんしね。無調の曲などは，ある音からある音までクオーターですべて鳴らす，なんていうものもあるくらいですから(楽譜上では黒ベタで塗ってあります)。

無調の曲については，いつか質問があってから，結構

勉強しました。機会があったら，どこかでまたお話ししてもいいかもしれませんね。

ところで，ちょっと話はずれますが，私はピカソの絵と，最近の目の大きなアニメ絵が非常に似ているように思うのです。どちらも現実からはかけ離れていますが，描いている人間の心をそのまま映したものとして評価できるのではないでしょうか。アニメ絵の女性の多くが目が大きいのは，たぶん絵描きの心には女性の目が美しく映ったのだろうし，ピカソの絵は見えない部分の美を表現しているという感じがするのです。私は絵描きではないから詳しくはわからないのですけれどね。

ピカソの絵は，音楽に直せば無調の曲って感じがして，よいと思うのです。絵のように，心に映ったものや感じたものをそのまま音楽にできたらいいですね。

§ドミナント進行による調の転移

さて，たった1つの調で淡々と曲を進めていっても，美しい曲を作ることができます。3コード(トニック，サブドミナント，ドミナント)だけでも，名曲は作れます。しかし，曲の単純な進行から逸脱したくなることは，作曲をしているとしばしばあります。実際，サビで転調する曲は，五万とありますものね。

これに対して，曲の構成の問題ではなく，ハーモニーを単調にさせないため，転調と同じようなことをするときがあります。つまり借用和音のことですが，この原理と転調の原理は同じようなものです。違いは何かというと，借用和音は近親調に転移したあと，あたかも近親調を「借用」したように，またもとの調へ移動して，そこに復帰するのに対して，転調は調が転移したあとに転移した先で解決を行います(厳密にはいえないけれど)。つまり転調の場合，完全に転移してしまって短期的には戻ってきません。

さて，我々がここで問題としているのは転調か借用かということではなく，実際に転調や借用はどうやって行われるかということです。ここで，ドミナント進行に対する知識が必要になってきます(ぎくっとした方は，先月号を参照してください)。

一般に，ドミナント進行はドミナント7から始まりドミナント6，ドミナント5，ドミナント4，ドミナント3，ドミナント2，ドミナント1と進み，最後にトニックで締めくくります。解説すると，ドミナント7にとってドミナント6はトニックであり，ドミナント6にとってドミナント5はトニックであり……と，従属してドミナント進行を繰り返していることがわかります(図1)。これはつまり，ある固有のスケールのなかに，潜在的にほかのスケールを隠しもっているといえます。つまりI以外の固有和音も，一時的な調の中心(トニック)となり得るというわけです。この法則は和声の性格ともいえるので，これを利用すると転調や借用がスムーズにいくことがわかりますよね?

たとえば，スケールはいまCだったとします。ここで2度上げてDに転調したい場合，DのドミナントはAですから，C→Dと一気に進むよりも，C→A→Dと進んだほうが，よりスムーズに進むことができます。

このような，ドミナント1～ドミナント7までの調を，近親調といいます。ただし，メジャースケールのⅦ，マイナースケールのⅡは，減3和音になるので，近親調になれません。

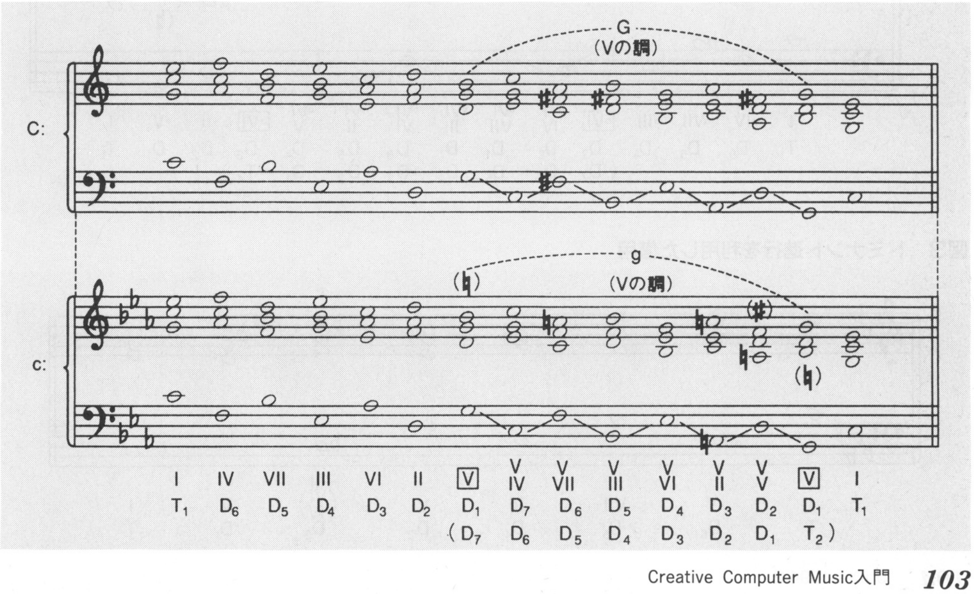
このようにドミナントモーションを利用すれば，近親調であるならば，作曲者は好きなときに好きな調へ転調することができます。

§旋法の変化

「転調」とは主音が変化するものをいいますが，実際には旋法(マイナースケールとか，メジャースケールとか)だけを変化させるものもあります。これを旋法転換と呼びますが，いいかえればこれは，単なる同主調への移動にすぎません。

主音の変化しない旋法転換は，考えたらすぐにわかるとおり(主音がCであれば)，Cm→CとC→Cmの2種類です。しかし，これは単なる机上の空論で，実際にできるのはメジャーからマイナーへの1方向です。これは音場のエネルギーがメジャーに比べマイナーのほうが小さいので，下方調転換という名目のもとに転換します。この法則は，音重力のところで説明しましたね。

図1　ドミナント進行



しかし，このままでは，いったんメジャースケールからマイナースケールに移行してしまったら，二度と原調へ戻れないということにもなりますよね。なんとか，ほかの調を迂回してでも原調へ戻ることはできないのでしょうか?

このようなときに，最初に頼りにしようと「思わなくてはいけない」のはドミナント進行です。ここでドミナント進行を思い出さなくては，いままで勉強した意味がなくなってしまいます。

図２を見てください。

まずトニックであるC(Ⅰ)から出発し，次のドミナント６(D_6)は旋法転換を利用し，マイナーに移行したとします。ここでマイナーに移行したということは，これはCm(Ⅰ)のドミナント６ということになり，このままではドミナント５もマイナーになってしまいます。このままCm上の固有和音を経て，ドミナント１まで進みますが，ドミナント１→トニックの進行だけは最後がメジャーに進行してもよくなります(なぜなのかは各自，考えましょう)。

このように，いったん短調へ移行してしまうと，最後(トニック)まで短調になってしまいますが，結果的にはD進行のように使うことができます。

§ 2次近親調

D進行を利用して近親調を借用できるといいましたが，この借用の最中にD進行を利用して借用することもできます(さらに旋法転換も兼ねて)。

図３を見てください。

極端な例ですが，これもまた借用を繰り返しているのがわかります。

まず，最初に旋法転換を兼ねてCmのD_6へ進行し，その途中にA♭の調を借用し，またCmの調へ戻り，原調へ復帰しています。

このように，いろいろな調を借用し，いろいろな調へ進むテクニックがわかったでしょうか?

§ 借用和音の追求

借用和音は，常に一時的なトニック(こういうのを副Tと表す)を意識して進行するのと同時に，原調への接続，関連を意識せねばなりません。これはすなわち，借用和音が，近親調としての機能と原調としての機能を併せもっていることを意味しています。借用和音の機能は，同じ音の固有和音(ダイアトニックトライアドコード)の機能をそのままもっているということになるわけです。

ということはつまり，借用された副Tの上でも，ドミナント進行(D進行)をするわけですが，同じようにコード進行は簡略化され，トニック(副T)，ドミナント(副D)，サブドミナント(副S)，ダブルドミナント(副D_2)という形をとることがあります。

さて，旋法転換された固有和音(準固有和音)以外の借用和音は，借用の借用といった２重の借用で原調に関連しています。ところが，なんらかのときに，こういった準固有和音が原調に直接関連をもつときがあります。

この場合，原調にはない音を借りてくるため，コードの構成音を半音上げたり，半音下げたりして丸く収めようとします。このようなものを，ナポリのⅡ，ドリアのⅣといいます。

図2　Ⅵを一時的なトニックとするドミナント進行

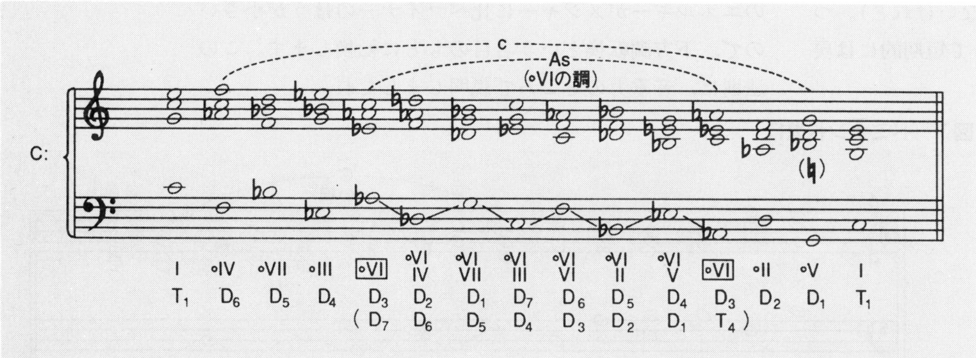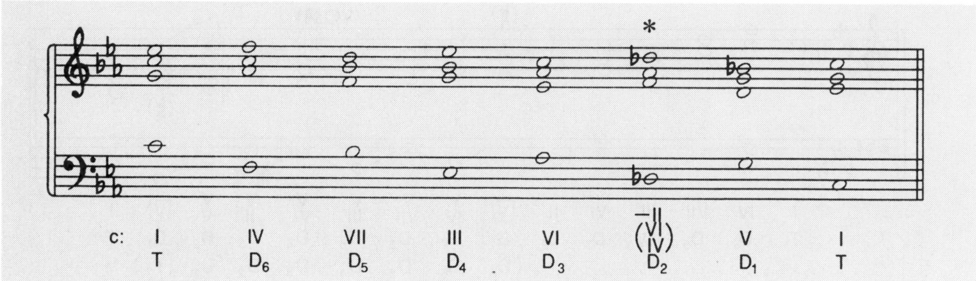


図3　ドミナント進行を利用した借用



●ナポリのⅡ

ダイアトニックトライアドコード，すなわち，固有和音には，どんな場合でも必ず１つの減３和音が含まれてしまいます。減３和音はたとえ一時的にでも副Tにはなりえないため，固有和音のなかでは最も弱い存在として考えられます。すなわち，このⅦは独立音度として利用することができません。

しかし，短調のⅡの場合，これは長調から求められたものであるため，長調のD進行から短調に旋法転換した場合，短調Ⅱは減３和音にもかかわらず，独立音度として利用することが可能です。しかし，この和音の欠点はそのまま残るのでなんとかして，うまく減３和音でなくしなくてはなりません。

そこで，根音であるIIの音を半音下げて−IIにすることにより，(IがCであればD♭)これはIImの代用として利用することが容易になります(図3)。−IIもほかの準固有和音のように副Tになりえます(図4)。

ナポリのIIはIがCとするならば，構成音はD♭，F，Aです。

●ドリアのIV

短調に旋法転換したとき，固有和音VI→VIIへの経過的進行は不可能になってしまいます(図5)。

この理由は明らかで，それはVIIが短調の場合，導音進行を重視するため，和声的にドミナントのVIIを半音上げてあり，その結果，S→D進行につけがまわってきて増2度進行するからです。

図4 −IIを一時的なトニックとするドミナント進行

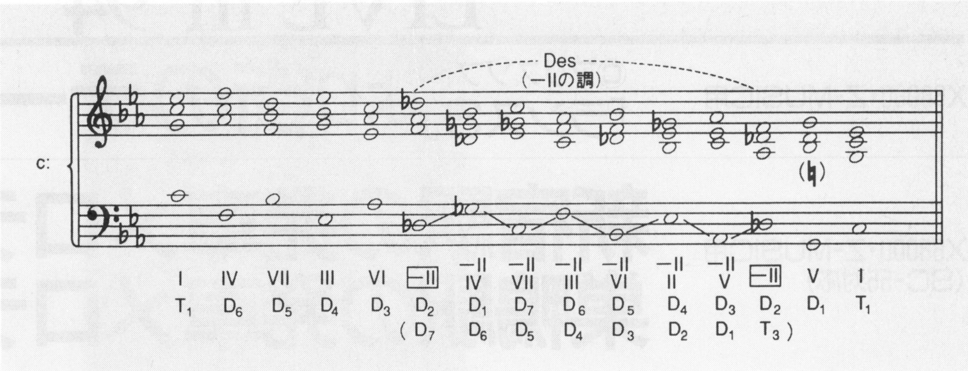
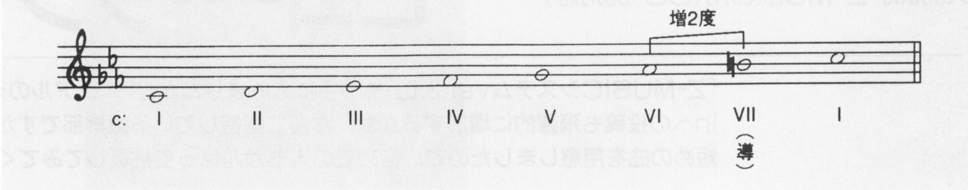


図5 短調に旋法転換したとき，VI→VIIの経過的進行は不可能

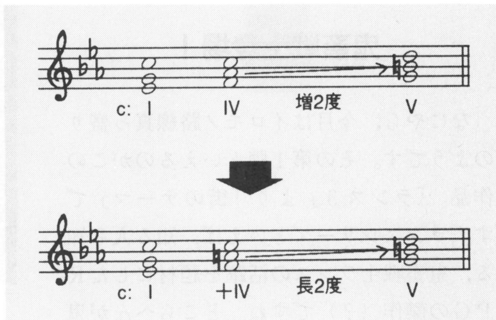


このため，VI音を半音高めて，S→D進行をより進みやすくするために，IVの代わりにドリアのIVといわれる音が使われます(＋IV)(図6)。

−IIと同様，＋IVも近親調ではありますが，近親調としては使われず，単にIVの代理として使われます。しかし，IVと同じように副Tになることができません。もちろん，この＋IVができた生い立ちからわかるように，この＋IVはVへ進む傾向があります。

構成音はIがCであった場合，FA<C(普通の短調はFA−<C)となります。

図6 IVの代わりに＋IV(ドリアのIV)を使う



§まとめ

とりあえず和声に関しての理論は，これで終了しました。長かったような短かったような気もしますが(いや，たぶんすごく長い)，ちゃんと内容を押さえてあれば(教え方がうまかったら?)，もはや和声に関してお手上げ状態になることは少なくなるはずです。

連載を始めてから，これで29回。なんだか結構，嘘というか，私自身の勘違いなどがあって，かえって読者を混乱させてしまったかもしれません。

最初に私がいっていたこと。

「最近，X68000にMIDIが普及しつつあって，パソコン通信などをしていると，何かのゲームミュージックのアレンジというものをよく見かけます。音源がグレードアップして，お気に入りのゲームミュージックを自分なりにアレンジしたいというのは，コンピュータミュージックが普及してきたいまとなっては，音楽好きの人にはあたりまえのことかもしれません。アレンジされたBGMを聴きながら思うのですが，センスがあるのにオクターバー(完全8度の音かぶせ，もう1オクターブ下の音などを重ねる奏法)しかアレンジの方法を知らないなど，もうちょっとアンサンブルを勉強していればいいのに，ということをしばしば感じるのです。そこで今回は，いかにしてうまくアレンジするか。という点について考えてみたいと思います」(連載第1回より)

なかなか偉そうなことをいってますなぁ。

初志貫徹できなかったことがたくさんあるような気がします。ごめんなさい。本を書くうえで途中から対象となる読者を変えてしまうなんて言語道断なのですが，連載を続けるうちに対象とする人が変わったこと変わったこと。最終的には，音楽の知識をもたない人よりも，むしろ音楽屋さんのほうに楽しく読んでもらって，コンピュータでサイドミュージックしたかった人たちには，ちょっと難しすぎたような気がします(単に説明が悪いだけのような気がしないでもないのですが)。

編集さんにも，よく迷惑をかけました(冷汗)。

連載中にも4人ぐらい担当者が変わったような……。そもそも，昔のOh!Xにこんなに楽譜が載ることはなかったと思いますし。

まあ，終わりがよいとは思えないけれど，とりあえずこれでCreative Computer Music入門は幕を閉じさせていただきます。

最後に。

読者を含め，私の隣人たちへ感謝の意をこめて，ここでひとまず終わりとさせていただきます。

Oh!X LIVE in '94

X68000・Z-MUSIC用

©アリスソフト「ランス3」より街のテーマ

Otani Kazutomo 大谷 一友

X68000・Z-MUSIC用（SC-55対応）

新宿駅の発車メロディ 巣鴨駅の発車メロディ

Yamada Kai 山田 開

X68000・Z-MUSIC用（SC-55対応）

ピコー・ソング

Yamada Kai 山田 開

「Z-MUSICシステムver.2.0」もう手に入れましたか。マニュアルの充実で，LIVE inへの投稿も飛躍的に増加するかな，なんて期待している編集部ですが……。今月は短めの曲を用意しましたので，初めての人もがんばって挑戦してみてくださいね。

鬼畜戦士登場！

なにやら，今月はイロモノ路線真っ盛りのようです。その第1弾といえるのがこの作品。「ランス3」より「街のテーマ」です。ランスシリーズといえば，知る人ぞ知る，鬼畜戦士ランスの活躍を題材にしたRPGの傑作（？）ですね。どこらへんが鬼畜なのかは実際にゲームをやったほうがわかりやすいですね。アリスソフトから発売されています。ピーンときた人もいるかもしれませんね。そう，Hゲームなのです。ちなみに，Z-MUSICシステムの作者，西川氏もけっこうハマって遊んでいました。

この作品は，題材がイロモノ（意味が違うか？）なだけで，そのほかは意外にも（失礼！）まっとうなのです。PCM8.Xも使っていないので，Z-MUSICシステムだけで楽しむことができます。

ランスで遊んだ人も，遊ばなかった人も，入力してランスワールドに浸ってね。

山田開，いきまぁすっ！

しつこいようですが，今月はイロモノなのです。あとの3曲は短いのでまとめてお贈りしましょう。いままで誰も目をつけなかった，駅の発車の合図シリーズ「新宿駅の発車メロディ」と「巣鴨駅の発車メロディ」，そしてCM曲では有名な「ピコー・ソング」です。3曲とも演奏にはZ-MUSICシステムとSC-55同等品が必要です。



ランス3

イロモノに解説をつけるのもナンセンスというものですが，あえて説明を加えておきましょう。

最近，駅の発車の合図はジリリリリ……というベルではありません。残念ながら私は「鉄っちゃん」（鉄道マニアor鉄道オタク）ではないので，詳しくは知りませんが，東京近郊のJRの駅ではショートフレーズの曲がかかります。駅ごとに違うメロディですが，同じ曲を使っている駅もあるようですし，上りホームと下りホームでは違うのかもしれません。詳しい方はぜひご一報ください。

今回登場した「新宿駅」と「巣鴨駅」は両方とも山手線の駅です。たぶん，新宿駅は埼京線の「武蔵浦和駅」と同じものだと思いますが……。

この作品は多少のアレンジが入っています。「巣鴨駅の発車」ではトラック1以外を消すとオリジナルになるようです。このことでもわかるように，アレンジ版とはいえ，もとの雰囲気はずいぶん残っています。ぜひ一度聴いてみることをお勧めします。

さてもう1曲は，あのサントリーの紅茶「ピコー」のCMのダンスでもおなじみになった「ピコー・ソング」です。あのCMは全国ネットでも流れていましたよね？　たしか，キャンペーン期間中にはがきを送ると，ダンスの解説つきの楽譜を手に入れることができたと思います。山田君もそれをもとにしたのかな？

いずれにせよ，どれも短くて楽しい作品ですので，軽い気持ちで入力して聴いてくださいね。

こんなに愉快な作品を投稿してくれた山田君ですが，実は一発屋（？）ではありません。以前（1993年10月号）にも「未来予想図II」が掲載されています。今回同封していただいた「未来予想図」もボーカルに進歩のあとがみられます。あとは全体のつながりを研究してみてくださいね。つながりの薄さが改善されれば，さらによくなるでしょう。ぜひまた投稿してください。彼は常連入りを目指して，月1投稿をしてくれていますが，今回の「イロモノ」も3回の投稿をひとまとめにしての発表です。まだ駆け出しの人も，この意気込みを見習ってみてはいかがでしょうか？　(SIVA)

リスト1 ランス3

```
/ 「 R a n c e III ～ リ ー ザ ス 陥 落 ～ 」
/                 " 街 の 曲 "
/      Programmed by Kazutomo Ohtani
/                           in 1992 8 26
.adpcm_block_data=rance3.zpd
/         AR DR SL SR RR TL KS MU D1 D2 AM           E.PIANO
(@1,      20, 9, 2, 6,10,42, 2, 2, 0, 0, 0
          31, 5, 5, 6, 7,52, 2,10, 0, 0, 0
          31, 7, 3, 5, 5,42, 0, 1, 0, 0, 0
          31, 9, 3, 7, 5, 0, 1, 2, 0, 0, 0
/         AL FB OM PN
           2, 7,15, 3 )
/         AR DR SL SR RR TL KS MU D1 D2 AM           E.BASS
(@2,      31,10, 7, 8, 2,33, 0, 0, 7, 0, 0
          21, 8, 8, 7, 5,23, 3, 7, 7, 2, 0
          31, 5, 6, 7, 1,37, 0, 0, 3, 0, 0
          31, 8, 6, 7, 5, 0, 0, 1, 7, 0, 0
/         AL FB OM PN
           2, 7,15, 3 )
/         AR DR SL SR RR TL KS MU D1 D2 AM           S.BASS
(@3,      27,14, 0, 7, 3,34, 0, 8, 0, 0, 0
          31,10, 0, 8, 6,54, 0, 2, 0, 0, 0
          31,19, 0, 5, 6,15, 0, 0, 0, 0, 0
          31, 2, 0, 9, 1, 0, 0, 0, 0, 0, 0
/         AL FB OM PN
           3, 7,15, 3 )
/         AR DR SL SR RR TL KS MU D1 D2 AM           HIHAT1
(@4,      31, 5, 6, 0, 5, 0, 0,15, 7, 1, 0
          31,17, 4,15,15, 0, 0, 3, 0, 2, 0
          31,20, 6, 3, 3, 0, 0, 1, 7, 2, 0
          31,25, 6,15,15, 0, 0, 7, 0, 3, 0
/         AL FB OM PN
           4, 5,15, 3 )
/         AR DR SL SR RR TL KS MU D1 D2 AM           HIHAT2
(@5,      31, 5, 6, 0, 5, 0, 0,15, 7, 1, 0
          31,10, 4, 7, 8, 2, 0, 3, 1, 2, 0
          31,20, 6, 3, 3, 0, 0, 1, 7, 2, 0
          31,25, 6, 7,10, 0, 0, 7, 0, 3, 0
/         AL FB OM PN
           4, 5,15, 3 )
/         AR DR SL SR RR TL KS MU D1 D2 AM           KEYBOARD
(@6,      21, 0, 0, 0, 0,27, 0, 8, 7, 0, 0
          18, 0, 0,15, 0, 2, 0, 4, 7, 0, 0
          21, 0, 0, 0, 0,26, 0, 8, 3, 0, 0
          18, 0, 0,15, 0, 4, 0, 4, 3, 0, 0
/         AL FB OM PN
           4, 7,15, 3 )
/         AR DR SL SR RR TL KS MU D1 D2 AM           SYNTHE BELL
(@7,      31, 7, 0, 0, 0,32, 0, 5, 7, 0, 0
          31,12, 4, 7, 5, 0, 0, 1, 7, 0, 0
          31, 7, 0, 0, 0,32, 0, 5, 3, 0, 0
          31,12, 4, 7, 5, 0, 0, 1, 3, 0, 0
/         AL FB OM PN
           4, 6,15, 3 )
(i)
(b0)
(o128)
(m1,1000)(a1,1)
(m2,1000)(a2,2)
(m3,1000)(a3,3)
(m4,1000)(a4,4)
(m5,1000)(a5,5)
(m6,1000)(a6,6)
(m7,1000)(a7,7)
(m8,1000)(a8,8)
(m9,1000)(aADPCM,9)
/        K E Y B O A R D ( 2 行 目 か ら 音 量 だ け 違 う )
(t1)     @6 v12 l8 q8 p3 @b00 @h48 @s4 @m30
(t1)     r1r1r1r1
(t1)     o2 [do] |:f1r<cq4dc>bgabq8f1v6f4r2.v12
(t1)     a1r<eq4fed>b<cdq8>a1v6a4r2.v12:|
(t1)     >|:b2<c2d2rdeff4ge4.c4f4.e&e2
(t1)     >b2rb<cd&d2rgfed4.e&e2&e2.r4:| [loop]
(t2)     @6 v10 l8 q8 p3 @b25 @h48 @s4 @m30 r16.
(t2)     r1r1r1r1
(t2)     o2 [do] |:f1r<cq4dc>bgabq8f1v4f4r2.v10
(t2)     a1r<eq4fed>b<cdq8>a1v4a4r2.v10:|
(t2)     >|:b2<c2d2rdeff4ge4.c4f4.e&e2
(t2)     >b2rb<cd&d2rgfed4.e&e2&e2.r4:| [loop]
/        B E L L
(t3)     @7 v11 l8 q8 p3 @b00 @h48 @s4 @m30
(t3)     r1r1r1r1
(t3)     [do] |:r1r1r4o5fc4gdar<c4>b4gad |r1r1rb<cd4>a4f&f4r4r2:|r1r1rab<c4>b<cd4f4e4dc>b
(t3)     |:4r1r1r1r1:| [loop]
/        E . P I A N O
(t4)     @1 v14 l8 q7 p2 @b00 @h48 @s4 @m30
(t4)     o5   |: v14cv6cv14cv6cv14>bv6bv14b<cv6cv14cv6cv14c>bv6bv14bv6b<:|
(t4)     [do] |:8v14cv6cv14cv6cv14>bv6bv14b<cv6cv14cv6cv14c>bv6bv14bv6b<:|
(t4)     v11 q4 |:gggg gggg gggg gggg fffe eeee fffe eeee
(t4)             gggg gggg gggg gggg ddde eeee fffe eeee:| [loop]
(t5)     @1 v14 l8 q7 p1 @b00 @h48 @s4 @m30
(t5)     o4   |: v14fv6fv14fv6fv14fv6fv14ffv6fv14fv6fv14ffv6fv14fv6f:|
(t5)     [do] |:8v14fv6fv14fv6fv14fv6fv14ffv6fv14fv6fv14ffv6fv14fv6f:|
(t5)     v11 q4 <|:dddd dddd dddd dddd cccc cccc cccc cccc
(t5)             dddd dddd dddd dddd >bbb<c cccc cccc cccc:|> [loop]
/        B A S S ( 2 行 目 か ら 同 じ )
(t6)     @2 v11 l8 q7 p3 @b00
(t6)     r1r2r4r8o3c16d16frr2r4rfc>fr<c16d16c16d16c16>f16
(t6)     [do] |:4fr<fr>f<ef>frf<f>frf<ef> fr<fr>f<ef>frf<f>fr<efc:|
(t6)     l16|:>gr<drcrd>grgb8<c8d8 gfg8dcd>grgb8<c8d8
(t6)     cr>gr<cr>b<c8>ee8frgr <cr>gr<crfe8c>g8frer
(t6)     gr<drcrd>grg<g8crd8 >gr<drcrdgrfgdc>gfg
(t6)     <cr>gr<cr>g<crc>b<c>agfg <cr>gr<cr>g<cr>ee8frgr:| [loop]
(t7)     @3 v14 l8 q7 p3 @b25
(t7)     r1r2r4r8o3c16d16frr2r4rfc>fr<c16d16c16d16c16>f16
(t7)     [do] |:4fr<fr>f<ef>frf<f>frf<ef> fr<fr>f<ef>frf<f>fr<efc:|
(t7)     l16|:>gr<drcrd>grgb8<c8d8 gfg8dcd>grgb8<c8d8
(t7)     cr>gr<cr>b<c8>ee8frgr <cr>gr<crfe8c>g8frer
(t7)     gr<drcrd>grg<g8crd8 >gr<drcrdgrfgdc>gfg
(t7)     <cr>gr<cr>g<crc>b<c>agfg <cr>gr<cr>g<cr>ee8frgr:| [loop]
/        H I H A T
(t8)     @4 v10 l8 q8 p3
(t8)     o3r1r2r8c16c16@5v14crr1r1
(t8)     @4 [do] |:64v16cv12c:|
(t8)     l16|:64v14cv10cv12cv10c:| [loop]
/        D R U M S
(t9)     @d1 v8 l8 p3 q8 o1
(t9)     r1r2r4r8c8c1r<p1cp3dp2e>p3r8c16c16d4
(t9)     [do] |:grdrcrdcccdrcrdr crdrccdrcrdrccdr
(t9)     crdrccdcrcdrccdr |crdrccdcrcdrccdr:|crdrccdcrcdrccd16d16r16d16
(t9)     |:grdr16cc16cdc crdr16cc16cdr
(t9)     |:3ccdr16cc16cdc crdr16cc16cdr:|:| [loop]
(p)
```

リスト2 ランス3用音色コンフィグファイル

```
.o1c =bass¥snapk.pcm
.o1d =snare¥slds.pcm,p2
.o1g =cymbal¥crsh.pcm,v70,mo1c

.o2c =tomtom¥elect_4.pcm,p4,v140
.o2d =tomtom¥elect_3.pcm,p4,v140
.o2e =tomtom¥elect_2.pcm,p4,v140
```

リスト3 ランス3用カウンタ表示

```
1:00000300 00001800   2:00000312 00001800   3:00000300 00001800   4:00000300 00001800
5:00000300 00001800   6:00000300 00001800   7:00000300 00001800   8:00000300 00001800
9:00000300 00001800
```

リスト4　新宿駅の発車メロディ

```
/-----------------------------------------------------/
/              S T A T I O N   M U S I C              /
/                                                     /
/                           新宿 …                    /
/                                                     /
/                   (C)Japan Rail Way                 /
/                                                     /
/               Arrange By. Kai  Yamada               /
/-----------------------------------------------------/
(i)
(b1)
(d0)
(m1,8000)(a1,1)
(m2,8000)(a2,2)
(m3,8000)(a3,3)
(m4,8000)(a4,4)
(m5,8000)(a5,5)
(m6,8000)(a6,6)
(m10,8000)(a10,10)
(m11,8000)(a10,11)
/-----------------------------------------------------
.roland_exclusive $10,$42={$40,$00,$7f,$00}
.sc55_v_reserve $10={5,4,2,2,2,5,0,0,0,4,0,0,0,0,0,0}
.sc55_reverb $10={4,4,0,90,50,0,0}
/-----------------------------------------------------
(t1)          @i$41,$10,$42
(t1)            i0  @9  o4  v16  l8  q8  p3  @e127,0  t140
(t1)          r4<g>gag<c>g<e>g<g>gag<c>g<e>g<g>
(t1)          gagbg<d>g<g>gagbg<d>ggeg<cccegv14'c2eg<c',6
/-----------------------------------------------------
(t2)          @i$41,$10,$42
(t2)            i0  @49  o4  v8  l1  q8  p3  @e127,80
(t2)          r4|:4'egc':|'eg<c''g2<ce'
/-----------------------------------------------------
(t3)          @i$41,$10,$42
(t3)            i0  @33  o2  v12  l1  q8  p3  @e127,0
(t3)          r4|:c2..c8c1:|g2e2c2
/-----------------------------------------------------
(t4)          @i$41,$10,$42
(t4)            i0  @1  o4  v12  l8  q8  @p104  @e127,10
(t4)          r4|:5g4.eceg4:|
/-----------------------------------------------------
(t5)          @i$41,$10,$42
(t5)            i0  @49  o6  v6  l8  q8  @p44  @e80,10
(t5)          r4|:5g4.ec4.e:|
/-----------------------------------------------------
(t6)          @i$41,$10,$42
(t6)            i0  @11  o5  v12  l8  q8  p3  @e127,40
(t6)          r4<g>gag<c>g<e>g<g>gag<c>g<e>g<g>
(t6)          gagbg<d>g<g>gagbg<d>ggeg<ccceg'c2eg<c',6
/-----------------------------------------------------
(t10)         @i$41,$10,$42
(t10)           i0  @1  o2  v14  l8  q8  p3  @e127,0
(t10)         @y$1a,42,$50@y$1a,46,$50
(t10)         @y$1c,57,$65@y$1c,49,$15
(t10)         @y$1a,57,$45@y$1a,49,$45
(t10)         @y$1c,45,$60
(t10)         r4c4dcc4'df''ca'c4dcc4'df'c
(t10)         c4dcc4'df''ca'c4dcdddd
(t10)         'c4<a'd'c4<c+'dc'd<a''d<c+'
(t11)         r4o2l8|:4|:7f+:|a+:|
(t11)         |:5a+4:|
/-----------------------------------------------------
(p)
.sc55_print "STATION MUSIC"
.comment    駅の発車合図より　～新宿バージョン（アレンジ）～
```

リスト5　新宿駅の発車メロディ用カウンタ表示

```
1:00000450 00000000   2:00000450 00000000   3:00000450 00000000   4:000003F0 00000000
5:000003F0 00000000   6:00000450 00000000  10:00000408 00000000  11:00000420 00000000
```

リスト6　巣鴨駅の発車メロディ

```
/-----------------------------------------------------/
/              S T A T I O N   M U S I C              /
/                                                     /
/                           巣鴨 …                    /
/                                                     /
/                   (C)Japan Rail Way                 /
/                                                     /
/               Arrange By. Kai  Yamada               /
/-----------------------------------------------------/
(i)
(b1)
(d0)
(m1,5000)(a1,1)
(m2,5000)(a2,2)
(m3,5000)(a3,3)
(m4,5000)(a4,4)
(m5,5000)(a5,5)
(m6,5000)(a6,6)
(m7,5000)(a7,7)
(m10,5000)(a10,10)
(m11,5000)(a10,11)
/-----------------------------------------------------
.roland_exclusive $10,$42={$40,$00,$7f,$00}
.sc55_v_reserve $10={4,2,4,2,2,4,2,0,0,4,0,0,0,0,0,0}
.sc55_reverb $10={4,4,0,90,50,0,0}
/-----------------------------------------------------
(t1)          @i$41,$10,$42
(t1)            i0  @12  o5  v15  l8  q8  p3  @e127,0  t100
(t1)          r4e>g<ced>gb<dc>ea<c>babg<e>g
(t1)          <ced>gb<dc>ea<c>bab<d>v12'c2eg<c',6
/-----------------------------------------------------
(t2)          @i$41,$10,$42
(t2)            i0  @33  o2  v11  l2  q8  p3  @e127,0
(t2)          r4c>bag<c>bab<c
/-----------------------------------------------------
(t3)          @i$41,$10,$42
(t3)            i0  @49  o4  v8  l2  q8  p3  @e127,80
(t3)          r4|:'eg<c''gb<d''ea<c''egb':|'ceg<c'
/-----------------------------------------------------
(t4)          @i$41,$10,$42
(t4)            i0  @1  o3  v8  l8  q8  @p34  @e80,10
(t4)          r4|:ge4cbg4dae4ce4g4:|
/-----------------------------------------------------
(t5)          @i$41,$10,$42
(t5)            i0  @7  o4  v8  l8  q8  @p94  @e100,0
(t5)          r4|:cegedgbgceaebgb|e:|ge
/-----------------------------------------------------
(t6)          @i$41,$10,$42
(t6)            i0  @49  o4  v10  l8  q8  p3  @e127,20
(t6)          r4e>g<ced>gb<dc>ea<c>babg<e>g
(t6)          <ced>gb<dc>ea<c>bab<d>'c4eg<c',6
/-----------------------------------------------------
(t7)          @i$41,$10,$42
(t7)            i0  @49  o6  v8  l2  q8  @p79  @e50,10
(t7)          r4c>bag<c>bab<c
/-----------------------------------------------------
(t10)         @i$41,$10,$42
(t10)           i0  @1  o2  v12  l4  q8  p3  @e70,0
(t10)         @y$1a,42,$60@y$1a,46,$60@y$1c,48,$65
(t10)         r4|:3cd8c8c'df':|cd8c8c8<c16>a16'fd<a''d<c+8a'
(t11)         r4o2|:|:15f+8:|a+8:|
/-----------------------------------------------------
(p)
.sc55_print "STATION MUSIC"
.comment    駅の発車合図より　～巣鴨バージョン（アレンジ）～
```

リスト7　巣鴨駅の発車メロディ用カウンタ表示

```
 1:00000390 00000000   2:00000390 00000000   3:00000390 00000000   4:00000330 00000000
 5:00000348 00000000   6:00000360 00000000   7:00000390 00000000  10:00000348 00000000
11:00000330 00000000
```

リスト8　ピコー・ソング

```
/----------------------------------------------------/
/                       P E K O E                    /
/                                                    /
/                   P E K O E   S O N G              /
/                                                    /
/                      (C)SUNTORY                    /
/                                                    /
/                          By. Kai  Yamada           /
/----------------------------------------------------/

(i)
(b1)
(d1)
(m1,8000)(a1,1)
(m2,8000)(a2,2)
(m3,8000)(a3,3)
(m4,8000)(a4,4)
(m5,8000)(a5,5)
(m6,8000)(a6,6)
(m7,8000)(a7,7)
(m8,8000)(a8,8)
(m9,8000)(a9,9)

/-----------------------------------------------------

.roland_exclusive $10,$42={$40,$00,$7f,$00}
.sc55_v_reserve $10={2,2,4,4,3,2,2,2,2,0,0,0,0,0,0,0}
.sc55_reverb $10={3,2,0,72,50,43,0}

/-----------------------------------------------------

(t1)          @i$41,$10,$42
(t1)            i0  @1  o4  v16  l16  q6  p3  @e127,0  t140
(t1)          r4f+8.f+f+8.eg4f+8.>a<f+8.f+f+8.dq8e4
(t1)          q6e8.ee8.ee8.eq8e4f+4q6g8.ee8.f+q8d4

/-----------------------------------------------------

(t2)          @i$41,$10,$42
(t2)            i0  @22  o5  v16  l16  q6  p3  @e127,10
(t2)          r4f+8.f+f+8.eg4f+8.>a<f+8.f+f+8.de4
(t2)          e8.ee8.ee8.ee4f+4g8.ee8.f+d4

/-----------------------------------------------------

(t3)          @i$41,$10,$42
(t3)            i0  @1  o3  v9  l16  q8  p3  @e50,80
(t3)          r4'd2f+a''g4b<d''d4f+a'
(t3)          |:'e8.f+a''af+e':|>'a2<c+e'|:'a8.<c+e''<ec+>a':|
(t3)          'a4<c+e'<'d4f+a''g4b<d'>'a4<c+e'<'d4f+a',4

/-----------------------------------------------------

(t4)          @i$41,$10,$42
(t4)            i0  @8  o4  v5  l16  q8  p3  @e90,80
(t4)          r4'a2f+d''g4b<d''d4f+a'
(t4)          |:'e8.f+a''ef+a':|>'a2<c+e'|:'a8.<c+e''a<c+e':|
(t4)          'a4<c+e'<'d4f+a''g4b<d'>'a4<c+e'<'d4f+a',4

/-----------------------------------------------------

(t5)          @i$41,$10,$42
(t5)            i0  @74  o4  v7  l16  q8  @p94  @e100,0
(t5)          r4d4a8.ab4{af+d}4a4{af+e}4>a4<e4
(t5)          c+2.d4e2d4

/-----------------------------------------------------

(t6)          @i$41,$10,$42
(t6)            i0  @71  o4  v8  l16  q8  @p14  @e70,10
(t6)          r4d4a8.ab4{af+d}4a4{af+e}4>a4<e4
(t6)          c+2.d4e2d4

/-----------------------------------------------------

(t7)          @i$41,$10,$42
(t7)            i0  @1  o4  v10  l16  q8  p3  @e127,30
(t7)          r4d8.dd8.de4d8.f+e8.ee8.ef+4c+8.c+
(t7)          |:c+8.c+:|c+4d4e8.c+c+8.c+f+4

/-----------------------------------------------------

(t8)          @i$41,$10,$42
(t8)            i0  @33  o2  v12  l16  q8  p3  @e127,0
(t8)          r4d2g4d4e2>a2<e2a4d4g4{aec+}4d4

/-----------------------------------------------------

(t9)          @i$41,$10,$42
(t9)            i0  @73  o5  v6  l16  q8  @p100  @e127,0
(t9)          r4r8.a<d8rf+g4r4>r8.f+a8r<de4
(t9)          r4r8.>a<c+8rea4r4g4>a4<d4

/-----------------------------------------------------

(p)

.sc55_print "PEKOE SONG"
.comment      サントリー・コマーシャルより　～ピコー・ソング～
```

リスト9　ピコー・ソング用カウンタ表示

```
1:00000300 00000000   2:00000300 00000000   3:00000300 00000000   4:00000300 00000000
5:00000300 00000000   6:00000300 00000000   7:00000300 00000000   8:00000300 00000000
9:00000300 00000000
```

いつもより狭いのでさっそくいきます。

★ランス3

イロモノなんていわれていますが，データは真面目。私はこの曲を知りませんが，バランスや構成は非常に素直で，音採りの間違いもなさそう(この曲は耳コピだそうで)。楽譜がなきゃコピーできないという人も，こういった簡単そうな曲を題材にして始めてはどうかな。

★駅の発車メロディ&ピコー・ソング

発車メロディは個人的に笑えました。曲の最後が強引なのもグー。でもこれってあまりに極地ネタですね。知らない人にはきつかったかも。

どのデータも，クオリティをもう少し上げるとさらにウケるんじゃないかな(短くてお手軽なのも捨て難いけど)。狙ったデータであればこそ，クオリティの高さも重要。それがさらなる笑いへ導いてくれるというものでありましょう(ホントか？)。

◇　◇　◇　◇

▼(進)の「ちょっといいですかぁ?」▲

ところでZ-MUSIC愛好者の皆さん，「ZAM.R」というツールをご存じですか？

これは一種のステイタスビュアで，表示パラメータの違いや画面構成の違いはありますが，MON.Rのビジュアル版だと思って差し支えありません。「ZMUSICシステムver.2.0」に収録されているので，使った人もいるでしょうね。

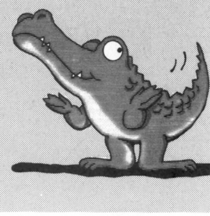
で，これがまた便利！　曲を作るときや，データの解析などにおいてかなりの威力を発揮します。私も「LAST WAVE」を作る際，コードワークや，リストじゃよくわからないメロディラインの確認のため，ずいぶんお世話になりました。通常必要だと思われる情報はほとんど表示されていますから，演奏データがどこで何をしているのかが一目瞭然。データ制作者にとって強力な助っ人となるでしょう。これは必携。

あなたが「コマンドの使い方がよくわからない」という初心者なら，これを使ってさまざまなデータを覗いてみてください。きっとなにかヒントが得られるはずです。

とにかく，これはイチ押しです。(進藤慶到)



(善)のゲームミュージックでバビンチョ



西川善司

●VIRTUA FIGHTER AKIRA/KAGE
SINGLE CD:TYDY-2056　1,000円(税込)
東芝EMIユーメックス　1/19発売

完全3Dポリゴンモデルによる格闘ゲームの「バーチャファイター」のBGMが早くもリリースされる。格闘ゲーム定番のナショナリティ路線かと思いきや，完全なダンサブルミュージックで意表を突かれた。KAGEのテーマは即興性が強くディストーションギターの単純なリフにアドリブメロディが展開するパターンで，トリッキーなリズムSEが曲の雰囲気を盛り立て，聴く者を熱くする。2月には同ゲームのビデオもリリースされる予定らしい。

お勧め度　10

●豪血寺一族/ATLUS
CD:PCCB-00145　1,500円(税込)
ポニーキャニオン　1/21発売

相手の精気を吸って若返るババアが妙にうけて一躍有名になった格闘ゲーム「豪血寺一族」。グラフィックやストーリー設定も特異だが，サウンドもかなり異端的なノリで度肝を抜いた存在となっている。純和風合の手が心地よい音頭調，素人っぽい掛け声やたるい手拍子による応援歌調，和風音律のもとに軽快に三味線が走る民謡調……といままでのゲームミュージックにはなかったジャンルを開拓している。肉声PCMを効果的に使用し，どの曲も面白い。全曲オリジナルサウンドで，ファン待望のS.E.&VOICEコレクションも完全収録だ。

お勧め度　8

●究極戦隊ダダンダーン
CD:KICA-7627　2,800円(税込)
キングレコード　1/21発売

コナミの脳味噌ぶっとびボンバー格闘アクションゲームのオリジナルサウンドアルバムがついに登場。近年PCM音源のクオリティとメモリ容量の増加でゲームミュージックもゴージャスになってきたが，このダダンダーンではある意味究極の形態に到達している。というのもステージ1のBGMはあの「泳げ！たいやき君」の子門真人がダダンダーンの主題歌を熱唱しているのだ。「いかにもヒーローもの」系の歌詞&子門ビブラートボイス。この取り合わせは古きよき70年代を思い起こさせる。2面以降のBGMも懐かしのワウワウギターがリズムを刻むは，山なりフレーズのピックベースが走るは，アクセントでティンパニやコンガが鳴るわで，当時のヒーローサウンドのノリを見事に再現している。でもコナミ節はちゃんと隠れていて単なる音ネタに終わっていないのがさすが。注目の1枚。

お勧め度　9

●DELICIOUS SELECTION/GAMADELIC
CD:PCCB-00143　2,500円(税込)
ポニーキャニオン　1/21発売

データイーストのゲームミュージックチーム「ゲーマデリック」のベストアルバム。いままでのゲームミュージックCDに収録されたアレンジバージョンのうち人気と完成度の高かったものを一挙にまとめたのがこの1枚。空牙やサンダーゾーン，ファイターズヒストリーなどのメジャータイトルはほとんど網羅。さらにゲーマデリックのテーマは新アレンジにて収録。

お勧め度　8

●ナムコゲームサウンドエクスプレス Vol.11「リッジレーサー」
CD:VICL-15025　1,500円(税込)
ビクターエンターテインメント　1/21発売

リアルタイムテクスチャマッピングで強烈なリアリティの表現に成功したポリゴンカーレースゲーム。こちらもBGMはメロディレスの曲が多い。メインテーマ「リッジレーサー」はなんとなくターボアウトランの曲のイメージだが，思わずアクセルを踏み込んでしまいそうな攻撃的なベース&リズムに乗せて暴れまわるナムコメロディはやはりセガとは違う風味。最近流行のジュリアナ系サウンドやDEATH-MIXタイプの曲もある。こっちは好みが分かれそうだ。

お勧め度　8

●SFC版交響組曲「ドラゴンクエストⅠ」
CD:SRCL2733　2,800円(税込)
ソニーレコード　発売中

社会現象にまで発展した同ゲームをファミコンからスーパーファミコン(SFC)へ，リメイク移植。それに伴いサウンドもアレンジされた。そのサントラであるこのCDは前半にオーケストラによる演奏，後半にオリジナルサウンドを収録している。さすがSFC，メモリ制約内でなんとかオーケストラサウンドを実現しようと頑張っているかいあって，ご家庭ゲーム機のゲームミュージックのなかではこの上ない完成度だ。曲はお馴染みのものを音源進化に伴いマイナーバージョンアップをしている。来月はSFC版「ドラクエII」のCDが発売予定。

お勧め度　9

●ツインビーPARADISE Vol.1
CD:KICA-7626　2,800円(税込)
キングレコード　1/21発売

ツインビーの世界がアニメ風のラジオドラマになり文化放送で毎週日曜0：00より「ツインビーPARADISE」というタイトルで放送中。この1～4話を収録。出演声優は山口勝平，田中真弓，國府田マリ子など。誰も死なない殺さないの憎めない悪役とドジな主人公たちがどんぶり島で暴れまわる。声のイメージや全体的なノリはゲーム世界のイメージそのままで好感度高し。初めから最後まで一定のクオリティの笑いが盛り込まれていてストーリーのレベルも高い。ツインビーファンは必聴。ただ，内容の性格上何度も聞けるものではないが。

お勧め度　8



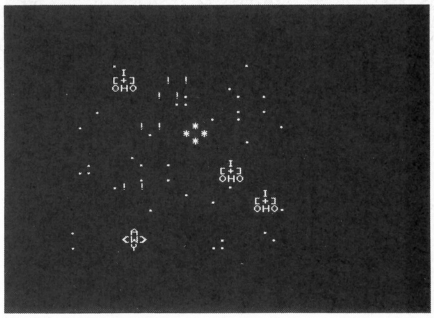
究極戦隊ダダンダーン

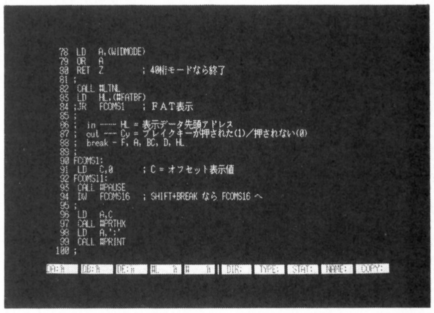


ドラゴンクエストⅠ

THE SENTINEL

〈対応機種一覧〉●MZ-80 K/C/700/1500●MZ-80 B/2000●MZ-2500/2861●X 1●X 1 turbo/Z●PC-8001/8801/88●SMC-777/C●PASOPIA/5●PASOPIA/7●FM-7/77/AV●MSX/2/2+/turbo R●PC-286/386/486/9801/98/9821●X 68000/X 68030
掲載されたプログラムの利用には各機種用のS-OS“SWORD”システムが必要です。





第140部　YGCSver.0.20ユーザーズマニュアル
第141部　S-OSで学ぶZ80マシン語講座(3)

●ようやく仕様公開

今月は，まず「YGCSver.0.20」リファレンスマニュアルをお届けします。

まずは，マニュアルのみですが，じっくり読んでみてください。一度でもゲームを作った経験があれば，読みこなせると思います。

また，経験がなくても「実際のゲームシステムはこうなっているんだ」という視点で読んでいただければ，ゲーム作成のコツというものがわかるでしょう。

確かに初めての人には，かなり重たいものだと思います。しかし，これを理解することによって，普段は見えない実際のゲームの中にあるシステムを，理解するチャンスなのですからがんばってみましょう。

そして，身についた知識はいずれ，必ず役に立ちます。

さて，システム自体は，共通化できる部分をなるべくシステムが請け負って，あくまでもゲームの個性を演出する部分（キャラクターの動きなど）をユーザーがプログラミングするようになっています。具体的にどの部分を用意すべきか，しっかり把握しましょう。

このシステムによって，ある程度ゲーム作成の手間が省けます。それでも，自分でつくらなきゃならない部分が多いじゃないか，と思う人がいるかもしれません。現在のシステムでサポートしているのは，本当に基本的なことです。ですから，足りないものがあるのは当然です（やろうと思えばこれでも十分だけど）。そう思ったなら，具体的にどのようなことをシステムでサポートしてほしいか，ご意見，ご希望をお寄せください。

ところで，1月号で募集した「YGCSver.0.20」のモニタですが，実はこの原稿を書いている時点で，まだ応募のハガキが見当たりません。さすがに，1月号の発売からさほど時間がたっていないということもありますが，もしかしたら，このまま……なんてことになったらどうしよう（ちょっとあせってしまう）。

ええい，こうなったら先着30名などと，けちくさいことはいわん。どど～んと無制限募集だ。やりたい人，全員に配布しようじゃありませんか。

しかし，機種，配布メディアについては1月号で紹介したものにかぎります。編集部にあるシステムで対処できるものでないとコピー作業ができませんので，ご理解のほどをお願いします。

では，引き続きご応募ください。

●マシン語講座

今月で4回目を迎えた「S-OSで学ぶZ80アセンブラ講座」。S-OSの扱い方，Z80を使ったプログラミングのコツが，徐々にわかってきましたでしょうか。

解説とリストを照らし合わせながら，がんばってリストを読みこなし，S-OS，そしてZ80を学んでください。

また，この講座で教えることがすべてではありません。とりあえず，基本的なことを話しているだけですので，もっと簡単で有効な方法もあるでしょう。

読者の皆さんが，記事を読んでなにか気になることがあれば遠慮なく突っ込んでやってください。

あと，MSX用S-OS“SWORD”を制作した筑紫氏から，Z80についていろいろなテクニックを調べ上げた原稿も届いています。普通のプログラミングに必要なテクニックからZ80の未定義命令まで言及した，なかなか読みごたえのある内容です。もともとは，立ち消えになっている「THE SENTINEL WORLD」への投稿だったのですが，本当にお蔵入りするにはもったいない原稿なので，いずれ，機会を狙って紹介したいですね。

いくら8ビット機とはいえ，せっかくのマシンをホコリを被らせたままにしておくのはもったいないことです。ぜひ，マシン語講座を活用してください。

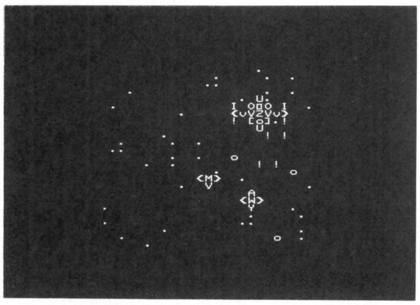
1994■インデックス

全機種共通
S-OS"SWORD"要

Yu-ri Game Core System
YGCS ver.0.20 リファレンスマニュアル

Aizawa Eiki
相沢 栄樹

まずは，仕様公開ということで「YGCSver.0.20」のリファレンスをお送りします。より完成されたシステムにするために，なにが足りないのかじっくり考えてみてください。



本システムは，S-OS "SWORD" 上でシューティングゲームを容易に制作するために開発されたシステムです。

ユーザーは本システムを使用することにより，ゲーム制作において避けて通ることのできない，画面制御，キャラクター制御などの煩わしい部分を考えることなく，アルゴリズムの記述に専念できるようになります。具体的にユーザーが制作すべきところは，

1)　ゲーム周りの処理（タイトル，ゲームオーバーなど）
2)　個々のキャラクターのプログラム
3)　キャラクターの出現スケジュール

など，大まかに3点に集約されます。

以上の3点において，1)の処理は完全にユーザーのオリジナルになりますが，残りの2点については決まったガイドラインに沿ってプログラムしてもらうことになりますので，注意してください。ガイドラインについては以降の項目で解説します。

それでは，このシステムの提供する機能について書き出しておきます。

1)　ゲーム全体の制御
2)　キャラクター，背景などの表示や制御(管理)
3)　キー入力の簡略化
4)　方向64段階，スピード256段階の移動制御
5)　64段階の方向サーチ
6)　加減速追尾制御（オプション用）
7)　画面外チェック
8)　当たり判定
9)　背景スクロール

以上，現バージョンにおいてサポートしている機能です。それでは，個々の解説をしていきます。

動作シーケンス

本システムの大まかな動作シーケンスを示しておきます。

1)　キャラクターの出現
2)　現在ワーク上に存在しているキャラクターの，個々の呼び出し
3)　全体の表示

それでは順番に説明していきます。

1)では，ユーザーが定義したスケジュールに従って，キャラクターを出現させていきます。出現のタイミングは，システムカウンタの値により制御され，システムカウンタ自体はここの処理が呼び出されるたびに，1ずつインクリメントされます。

2)では，各キャラクターのもっているIDコードにより，個々のプログラムを呼び出します。IDコードと個々のプログラムの対応は，ユーザーが定義します。当たり判定もここで行います。

3)では，プレイヤー，敵，背景の各スクリーンを合成して表示します。

大まかではありますが，以上が本システムの動作シーケンスです。本システムを使用する際は，このシーケンスを念頭に置いてください。

プログラミングガイド

本システムの守らなければいけない取り決めを示しておきます。

1)　本システムは，$3000番地に常駐させます
2)　ユーザーエリアは$6000番地以降を使用してください（ver.0.20現在）
3)　ユーザープログラムからは，システムコールを使用して各機能を呼び出してください。決して内部ルーチンを直接呼び出したり，内部ワークを直接参照するようなことをしないでください（一部例外あり）
4)　ユーザープログラムでは，システムで決められたフォーマットを必ず守ってください。そうでないとシステムが正確に動作できなくなります
5)　ユーザープログラムでは，システムによって提供されるワーク以外はできる限り使用しないでください。なぜならば，ユーザープログラムは多重呼び出しされることが前提になっているからです
6)　本システムでは，裏レジスタを退避エリアなどに使用しています。もし裏レジスタを使用することがあれば，自己の責任において対処してください
7)　本システムのブートストラップについては，決められた手順があります。手順に従わずに使用すると，システムが誤動作(暴走など）する可能性があります。注意してください
8)　本システムではIXレジスタをシステムポインタとして使用しています。決して値を変更しないでください

以上の8項目は必ず守ってください。それでは，順を追って説明していきます。

システムの起動

システムの起動（ブートストラップ）は以下の手順で行ってください。

○システム全体の初期化
　COLD_

リスト1　起動手順

```
        CALL    COLD_           ; System init
        LD      HL,CHR_TBL
        CALL    CHRF_           ; CHR ID table address set
        LD      HL,CHR_GEN
        CALL    GENF_           ; CHR generate table address set
        LD      HL,KEY_TBL
        CALL    KEYF_           ; Key table address set
;       LD      HL,STR_TBL
;       CALL    STRF_           ; Work structure table address set
        CALL    HOT_            ; Screen init & etc...
```

リスト2　基本制御のコーディング例

```
MAIN_LOOP:
        CALL    _BRKEY          ; 'BREAK' check
        JR      Z,EXIT
        CALL    PUSC_           ; Pause check
        JR      Z,MAIN_LOOP
        CALL    MAIN_           ; Game main
        LD      A,(PL_FLAG)
        OR      A               ; Player dead check
        JP      M,PLAYER_DEAD
        JR      MAIN_LOOP
```

○各種テーブルの登録
　　GENF_,CHRF_,KEYF_,(STRF_)
○システム起動準備
　　HOT_

リスト1に起動の手順のコーディングを示します。ここで「各種のテーブル」とありますが，これについてはのちほど説明します。

基本的なシステム制御

基本的な制御のコーディング例をリスト2に示しておきます。

リスト2を見ればわかるとおり，まずBREAKキーチェックをしています。これは強制的にプログラムを終了させるときに必要です。その次はポーズの処理です。PUSC_はポーズのON/OFFをチェックしています。もしポーズがONならば，なにもせずにループのトップへ戻ります。

MAIN_がこのシステムのジョブコントロールを受けもっています。ユーザーはこのシステムコールを呼ぶだけで，ゲームの処理の大半を実現できます(楽ですね)。当たり前のことですが，メインループを抜けるには条件がいります。一例としてプレイヤーの死亡チェックを入れてあります。実際には，ユーザー側で適当なチェックプログラムを入れておいてください。

最後はループの先頭に戻って終わりです。ユーザー側で必要なプログラムなどのパーツを用意しておけば，実際これだけでゲームは動きます。

各種テーブルの説明

●出現スケジュールテーブル

このテーブルにはキャラクターが出現するための情報が書かれています。フォーマットは表1のとおりです。

表1　出現スケジュールテーブル

```
    0 (2) : Generate count
    2 (1) : CHR number
    3 (1) : Search count
    4 (2) : Offset address
    6 (1) : X position
    7 (1) : Y position
例)
CHR_GENE_TBL:
;         COUNT:COD:SRCHCNT:OFADR:XP:YP
    DW      $0000,$00+01*$100,$0000,12+28*$100 ; Player set
                  :
    DW      $FFFF,$FF+00*$100,$0000,00+00*$100 (エンドデータ)
COUNT    : 出現カウント
COD      : キャラクターＩＤ
SRCHCNT  : ワークサーチ数
OFADR    : ワークサーチオフセット
XP       : Ｘ座標
YP       : Ｙ座標(*$100 は補正値なので無視してください)
```

これだけではわかりにくいので，テーブル内容をもう少し詳しく説明します。

・出現カウント

キャラクターが出現するためのカウント数。表1の例では0が設定されているので，スタート直後に出現します。

・キャラクターID

このキャラクターが何番のプログラムに対応するかが指定されます。表1の例では0番のプログラムが指定されています。

・ワークサーチ数，ワークサーチオフセット

ここは非常に重要です。本システムではキャラクター用ワークを64本もっていますが，何番から何番までが何用とは規定していません。よって一律平等に扱われますが，便宜上ワークを何等分かに分ける必要があります。ワーク構成例を表2に示しておきます。

では，表2のワーク構成例を基に考えてみましょう。単純に考えてワークサーチ数は，上記のサイズの数値を書けばよいわけです。よって，表1の例で出現させようとしているのはプレイヤーなので，ワークサーチ数は1となります。次にワークサーチオフセットですが，本システムにおいてキャラクターワーク1本につき32バイトが割り振られています。よって，次のような式で計算してください。

　　ワークサーチオフセット＝
　　　　　　　　スタートポイント×32

そうすると，表1の例ではプレイヤーワークに定義したいわけですから，

　　0×32＝0($0000)

となります。もし敵ワークに定義したいときは，

　　ワークサーチ数：16
　　16（ワークサーチオフセット)×32＝
　　　　512 ($0200)

とすればいいわけです。

表2　ワーク構成例

start	end	size	name
0	0	1	Player
1	15	15	Player bullet
16	31	16	Enemy
32	47	16	Enemy bullet
48	63	16	etc..

・X座標，Y座標

これは，出現するときの座標を指定します。表1の例では(12, 28)に設定されています。以上が出現スケジュールテーブルの書き方です。しっかりと理解しておいてください。あっ，忘れてましたがテーブルの最後は必ずエンドデータを置くようにしてください。

●キャラクターIDテーブル

これはそのままです。リスト3を見ればわかると思います。ID0番から順にそれぞれのキャラクターメインルーチンのアドレスを登録してください。

●キーテーブル

本システムでは，ジョイスティック入力を疑似的にサポートしています。そのため

には，ジョイスティックの入力をキーに割り当てなければなりません。その割り当てを定義するのが，このテーブルです。定義内容は表3のようになっています。

表3の各項目は各1バイトです。実際の書き方は定義例を参照してください。このデータを利用するには，システムコールKEYG_を使用することになります。このルーチンをコールすると，AレジスタおよびBレジスタに表4のフォーマットでデータが返ってきます。この機能によりキー配列が気にいらない人は，このテーブルを変更するだけで好みのキーに変えられます。よって，このシステム上でプログラムを作る場合は，できるかぎりシステムコールKEYG_を利用しましょう。

●ワーク構造テーブル

このテーブルは，キャラクターワークの構造を定義するためのテーブルです。「キャラクターワークの構造なら，すでに読んだ記憶があるぞ？」といわれる方，正解です。出現スケジュールテーブルのところで解説しています。

しかし，前のときは自分でオフセットアドレスを計算で求めていました。それではなぜ，わざわざ定義するのでしょうか？実はこのテーブル，あってもなくてもかまいません。このテーブルを参照するシステムコールは，現在1個しかありません（TNYS_）。よって，そのシステムコールを利用しない場合は，定義する必要がないわけです。テーブル内容の説明については，システムコールSTRF_のところを参照してください。

以上で各種テーブルの説明を終わります。本システムでは，これらのテーブルを参照することによってゲームを動かしています。使いこなすのはちょっと難しいでしょうが，がんばって理解してください。

ユーザープログラムガイド

本システム上でプログラムを走らせるにはいくつかの制約がありますが，これから書くことは規則ではなく，あくまでも「こう書いたほうがいいですよ」というプログラミングモデルです。ですが，中には必ず守らなければならないことも書いてありますので，飛ばさずに読んでください。

はじめにシステムの動作についてもう一度書いておきます。本システムは，ユーザーが定義したスケジュールに従いユーザーが定義したプログラムを呼び出す，といった動作でゲーム処理を実現しています。よって，ゲームのアルゴリズムに関する部分はすべて，ユーザーに任されることになります。それではユーザープログラムの解説に移ります。

○キャラクターワーク構成

まずはキャラクターワーク構成（表5）を説明します。

表5で，ユーザーワーク以外はシステムが参照するので，用途を変更しないでください。ワークの各項目を参照するときは，

LD　　A,(IX+FLG)

のように，必ずIXレジスタを通して参照してください。これは，ユーザーワークを参照するときも同じです。ユーザープログラムは，ユーザーワーク8バイト以内で動くように組まなければなりません。それ以上のワークを必要とする場合は，各自の責任において対処してください。基本的に，多重呼び出しに対応していればいいです。

それでは，各項目の説明をします。

◆FLG：キャラクターサイズ

00 : This work is not use.
01 : 1×1　PAT=chr code
02 : 2×2　PAT=pattern-A address
03 : 3×3　PAT=pattern-A address
04 : n×n　PAT=pattern-B address
-1 : No display mode
Pattern-A=Data only

例) DB ' A '　；3×3の例
DB '<M>'
DB ' Y '
Pattern-B = Size ＋ data

例) DB 5,4　　；5×4の例
DB ' I '
DB ' (V) '
DB '<WロW>'
DB ' o¥o '

リスト3　キャラクターIDテーブル

```
CHR_ID_TBL:
;          Label               ID   Name
   DW      PLAYER           ;  0 : Player
   DW      PL_BULLET        ;  1 : Player bullet
   DW      ENEMY_A          ;  2 : Enemy A
   DW      ENEMY_B          ;  3 : Enemy B
   DW      EN_BULLET        ;  4 : Enemy bullet

```

表3　キーテーブル

```
 0 : up            : Direction key
 1 : up-right      :     "
 2 : right         :     "
 3 : down_right    :     "
 4 : down          :     "
 5 : down_left     :     "
 6 : left          :     "
 7 : up_left       :     "
 8 : button A      : Button key
 9 : button B      :    "
10 : button C      :    "
11 : button D      :    "
定義例)
KEY_TBL:
     DB  '89632147'   ; Direction
     DB  'ZXCV'       ; Button
```

表4　KEYG_の返り値

```
ビット
  0 : UP (FORWORD)
  1 : DOWN (BACK)
  2 : RIGHT
  3 : LEFT
  4 : TRIGGER A
  5 : TRIGGER B
  6 : TRIGGER C
  7 : TRIGGER D
Aレジスタ：現在押されているキー
Bレジスタ：今回押されたキー
注) 出力されるデータはビットマップです。
```

表5　キャラクターワーク構成

```
1chr work size 32byts

OFS SIZ  NAME  NOTE
  0(1): FLG   : 00 - no use
                01 - 1x1 chr
                02 - 2x2  "
                03 - 3x3  "
                04 - nxn  "
  1(1): ATR   : 0 - player
                1 - option
                2 - pl bullet
                3 -
                4 -
                5 -
                6 -
                7 - 1/enemy
   2(1) : COD: CHR number
   3(1) : CNT: counter
   4(2) : XPS: X position
   6(2) : YPS: Y position
   8(2) : XDF: X difference
  10(2) : YDF: Y difference
  12(2) : PAT: pattern address
               (FLG = 01
                     CHR code)
  14(1) : MOD: move mode
  15(1) : ***:
  16(1) : DIR: direction
  17(1) : SPD: speed
  18(1) : POW: power
  19(1) : HPT: hit point
  20(1) : SXP: Start X pos
  21(1) : SYP:   "    Y pos
  22(1) : EXP: End X pos
  23(1) : EYP:  "   Y pos
  24-31 : User work (free use)
```

◆ATR：キャラクター属性
bit 0 - player
 1 - option
 2 - bullet
 3〜6 ユーザー定義
 7 - 0/player 1/enemy

ビット7が0のときはプレイヤー属性，1のときは敵属性です。同じ属性同士ではヒットチェックをしません。そのほかのビットはヒットチェック属性で，ヒットする者同士の同じビットが立っていると当たり。そうでないと通り抜けます。

例)
 No hit check
 &B00000100 <> &B00000100
 Hit check
 &B00000001 <> &B10000111
 No hit check
 &B00000010 <> &B10000100

◆COD：キャラクターIDコード
このコードによりユーザープログラムが呼ばれます。

◆CNT：カウンタ
このキャラクターが出現してからの時間をカウントしています。キャラクター出現時に0が代入されます。

◆XPS：X座標
X座標を保持します。1バイト目が実座標，2バイト目が小数点を保持しています。

◆YPS：Y座標
内容はX座標と同じです。

◆XDF：X移動差分
サインコントロールで使用します。

◆YDF：Y移動差分
内容はX移動差分と同じです。

◆PAT：キャラクターパターンアドレス
FLG を参照してください。

◆MOD：キャラクター処理モード
この値により処理を切り換えます。ただし，以降で示すスケルトンを使用したときにかぎります。キャラクター出現時は，常に0が代入されてます。

◆DIR：移動方向
サインコントロールで使用します。64段階でひと回りします。

◆SPD：移動スピード
サインコントロールで使用します。256段階の設定ができます。

◆HPT：ヒットパワー
ヒットしたときに，相手に与えるダメージを設定します。

◆POW：ライフポイント
基本的に，この値がマイナスになると爆発とみなします。ただし，その管理はユーザープログラムで行ってください。

◆SXP,SYP,EXP,EYP：ヒットサイズ
ここでヒット範囲の左上と右下を指定します。

◆User work：ユーザーワーク
ユーザープログラムで自由に使用してかまいません。

例) LD (IX+24),$20

以上，キャラクターワークの構成でした。

ユーザープログラムスケルトン

まずはスケルトンの例をリスト4に示します。

このスケルトンでは，処理モードによっ

リスト4 スケルトンプログラム

```
; ==============================
;              EXAMPLE
; ==============================
EXAMPLE:
    LD       HL,EXAMPLE_TBL
    JP       MODC_                 ; Mode control
EXAMPLE_TBL:
    DW       EX_MODE_0             ; 0 : Example init
    DW       EX_MODE_1             ; 1 : Example mode 1
;
;          <<< EXAMPLE INIT
;
EX_MODE_0:
    LD       (IX),?                ; CHR size
    LD       (IX+ATR),?            ; Hit check mode
    LD       HL,?                  ; CHR pattern set
    LD       (IX+PAT),L
    LD       (IX+PAT+1),H
    LD       (IX+POW),?            ; Hit power set
    LD       (IX+HPT),?            ; Hit point set
    LD       (IX+SXP),?            ; Hit size box
    LD       (IX+EXP),?            ;      "
    LD       (IX+SYP),?            ;      "
    LD       (IX+EYP),?            ;      "
;-----------------------
;    User init program
;-----------------------

    LD       (IX+MOD),1            ; Cange move mode 1
    RET                            ; Return system
;
;          <<< EXAMPLE MODE 1
;
EX_MODE_1:
;-----------------------
;    User CHR program
;-----------------------
RET                             ; Return system
; ==============================
```

注）ユーザープログラムを抜けるにはRETを使います。

表6 YGCSver.0.20コールアドレス一覧

NAME	ENTRY ADDRESS	
;＊＊＊＊＊System call		
COLD_	$3000	; System cold init (all init)
HOT_	$3003	; System hot init
MAIN_	$3006	; System job control
GENF_	$300F	; Character generate table address set
CHRF_	$3012	; Character table address set
VADR_	$3018	; VRAM address get
CADR_	$301B	; CHR work address get
PUSC_	$301E	; Pause check
RNDG_	$3021	; Random value generate
KEYF_	$3024	; Key list address set
KEYG_	$302D	; Key data get
;＊＊＊＊＊Service call : VRAM(BG)		
VRAMT_	$3040; SYS	; Vram display (all pages)
VRAMC_	$3043	; Vram clear (page:0-3)
VRAMF_	$3046	; Vram fill (page:0-3)
VRAMS_	$3049	; Vram scroll (page:3)
VRAMA_	$304C; SYS	; Vram address calc
;＊＊＊＊＊Service call : OBJECT		
OBJI_	$3060; SYS	; Object work init (all)
OBJD_	$3063; SYS	; Object display (all)
OBJC_	$3066; SYS	; Object clear (all)
OBJS_	$3069	; Object set
OBJF_	$306C	; Object free count
STRF_	$306F	; Work structure address set
TNYS_	$3072	; Tiny object set
;＊＊＊＊＊Service call : CHARACTER		
CHRG_	$3080; SYS	; Character generate
CHRC_	$3083; SYS	; Character control
HITI_	$3086; SYS	; Hit check init
HITC_	$3089; SYS	; Hit check
MODC_	$308C	; Mode control
SINC_	$308F	; Sin calc
MOVE_	$3092	; Move difference
SINM_	$3095	; Sin move
DIRS_	$3098	; Direction search
OUTC_	$309B	; Screen out check
OPTM_	$309E	; Option move

注）SYSがついているものは，基本的にシステムが内部で利用するためのシステムコールです。使用する場合には十分注意してください。

てプログラムを切り換える方法をとっています。まずモード0は初期化処理と固定して使いましょう。そのほかのモードはユーザーが自由に使用できます。たとえば，
<プレイヤープログラム>
　モード1：ステージスタート
　モード2：通常
　モード3：ステージクリア
　モード4：爆発
のように定義しておけば便利です。さらに使い方によっては，もっと便利な使用法もあります。いろいろ工夫してみてください。

モードを増やすには，モードテーブルに書き足すだけです。モードの切り換え方法は，次のとおりです。

　LD　(IX+MOD),モード番号

これで，次回の呼び出しからモードが変わります。

VRAMについて

VRAMの表示エリアは左上を(6, 7)とするサイズ(26, 25)の範囲です。VRAMの実画面エリアは，32×32のサイズが定義されています。

よって，上下に消えるキャラのYサイズは7キャラ以内，左右に消えるキャラのXサイズは6キャラ以内にそれぞれ収めないと，反対側から回り込んできて見えてしまいます。注意してください。

以上プログラミングガイドでした。ここに書いてあることはほんの基礎的なことだけです。今回はサンプルプログラムおよび，実際のシステムを発表できませんが，リファレンスを読んでだいたいの感じをつかんでください。

YGCSver.0.20簡易リファレンス

```
●COLD_
  in      : non
  out     : non
  break   : all
 システム全体の初期化をする。
●HOT_
  in      : non
  out     : non
  break   : all
 ラウンドスタート時に呼び出すことにより，画面やシステムカウンタなどを初期化します。
●MAIN_
  in      : non
  out     : non
  break   : all
 ゲームを実行します。
●GENF_
  in      : HL=table address
  out     : non
  break   : non
 キャラクター出現スケジュールテーブルの登録を行います。
●CHR_
  in      : HL=table address
  out     : non
  break   : non
 キャラクターIDテーブルの登録を行います。
●VADR_
  in      : A=page number
  out     : HL=VRAM address
  break   : AF
 VRAMの指定ページのアドレスを計算します。
●CADR_
  in      : L=Chr number
  out     : HL=Work address
  break   : AF,HL
 指定のキャラクターワークのアドレスを計算します。
●PUSC_
  in      : non
  out     Zero flag on=pause
  break   : AF
 一時停止の状態をチェックします。
●RNDG_
  in      : non
  out     : A=Random value
  break   : AF
  note    : I cycle is 245 counts.
 簡単な乱数の値を返します。245回で値が繰り返されます。
●KEYF_
  in      : HL   =Key list address
                  +0 - U   key
                  +1 - U_R key
                  +2 - R   key
                  +3 - R_D key
                  +4 - D   key
                  +5 - D_L key
                  +6 - L   key
                  +7 - L_U key
                  +8 - button A key
                  +9 -    "   B key
                  +A -    "   C key
                  +B -    "   D key
  out     : non
  break   : non
 キーテーブルの登録を行います。
●KEYG_
  in      : non
  out     : A   =Key data
               bit 0 - Up
                   1 - Down
                   2 - Left
                   3 - Right
                   4 - Button A
                   5 -    "   B
                   6 -    "   C
                   7 -    "   D
            B   =Key on edge data
               bit 0 - Up
                   1 - Down
                   2 - Left
                   3 - Right
                   4 - Button A
                   5 -    "   B
                   6 -    "   C
                   7 -    "   D
  break   : AF
 Aレジスタに現在押されているキーのデータが，Bレジスタに今回押されたキーのデータがそれぞれ戻されます。
●VRAMT_
  in      : non
  out     : non
  break   : AF,BC,DE,HL,BC',HL'
 VRAMをすべて合成して表示します。
●VRAMC_
  in      : A   =Clear page
               bit 0 - VRAM0
                   1 - VRAM1
                   2 - VRAM2
                   3 - VRAM3
  out     : non
  break   : AF
 指定のVRAMを初期化します。
●VRAMF_
  in      : A   =Clear page
               bit 0 - VRAM0
                   1 - VRAM1
                   2 - VRAM2
                   3 - VRAM3
  out     : non
            B=Fill chr code
  break   : AF
 指定のVRAMを指定のデータで埋めます。
●VRAMS_
  in      : A=0/V 1/H
  out     : non
  break   : non
 BGを指定の方向(縦/横)にスクロールします。
注) 現バージョン(ver.0.20)では縦スクロールにしか対応していません。
●VRAMA_
  in      : HL=Base VRAM address
  out     : non
  break   : AF,DE
 現在のキャラクターの位置に対応するVRAMのアドレスを計算します。VRAMのベースアドレスも指定してください。
●OBJI_
  in      : non
  out     : non
  break   : non
 キャラクターワーク初期化
●OBJD_
  in      : non
  out     : non
  break   : AF
 キャラクターをすべてVRAMに表示します。
●OBJC_
  in      : non
```

```
    out         : non
    break       : AF
```
キャラクターVRAM(プレイヤー/敵)を初期化します。

●OBJS_
```
    in          : A  =CHR ID number
                  B  =Search count
                  DE =Offset address
    out         : HL=Work address
                  Carry on=No work
    break       : AF,BC,DE,HL
```
キャラクターワークの指定範囲を調べてキャラクターを定義します。ワークに空きがない場合はキャリーフラグを立てて返ります。
注）範囲指定はキャラクター出現テーブルの項を参照してください。
Bレジスタ：ワークサーチ数
DEレジスタ：ワークサーチオフセット

●OBJF_
```
    in          : B=Search count
                  DE=Offset address
    out         : A=Count
    break       : AF,BC,DE
```
キャラクターワークの指定範囲を調べてワークの空き数を返します。
注）範囲指定はキャラクター出現テーブルの項を参照してください。
Bレジスタ：ワークサーチ数
DEレジスタ：ワークサーチオフセット

●STRF_
```
    in          : HL=Table address
    out         : non
    break       : non
```
ワーク構造テーブルを定義します。
<ワーク構成例>
```
 start end size   name
   0 -  0 :  1 : Player
   1 - 15 : 15 : Player bullet
  16 - 31 : 16 : Enemy
  32 - 47 : 16 : Enemy bullet
  48 - 63 : 16 : etc..
```
上記の構成例をもとに解説します。
1バイト目に項目数，次の2バイト目から項目データです。項目データは以下のとおりです。
1バイト目：何キャラ目から項目が始まるかを示します。
2バイト目：何キャラ分あるかを示します。項目データは項目数分用意してください。
上記構成例をコーディングしてみます。
```
WSTR_TBL:
; * * * 項目数
    DB      5
; * * * 項目データ
    DB       0,1    ; 0 : Player
    DB       1,15   ; 1 : Player bullet
    DB      16,16   ; 2 : Enemy
    DB      32,16   ; 3 : Enemy bullet
    DB      48,16   ; 4 : etc..
```
ちなみに項目番号は0番から始まります。

●TNYS_
```
    in          : A=CHR ID number
                  B=Table number
                  C=User data (24)
                  D=X offset
                  E=Y offset
                  H=Direction
                  L=Speed
    out         : non
    break       : non
```
指定の項目(Table number)に空きを探してキャラクターをセットします。このコールでは座標に，呼び出したキャラのXY座標にDEレジスタのオフセットを足した値をセットし，さらに以下のデータも同時にセットします。
C：ユーザーデータ1バイト（IX＋24）
H：方向（IX＋DIR）
L：スピード（IX＋SPD）
注）このコールではワークに空きがなかった場合でも，呼び出し側に知らせません。注意してください。OBJS_に比べて取り扱いが簡単になっていますが，その分処理速度を消費します。場合に応じて使い分けてください。STRF_でワーク構造テーブルを定義していない場合，動作の保証はありません。必ずワーク構造テーブルを定義してから使用してください。

●CHRG_
```
    in          : non
    out         : non
    break       : AF
```
出現スケジュールに従ってキャラクターを出現させます。このコールを呼ぶたびに，システムカウンタがインクリメントされます。

●CHRC_
```
    in          : non
    out         : non
    break       : ALL
```
IDに従って順番にユーザープログラムを呼び出します。

●HITI_
```
    in          : non
    out         : non
    break       : AF
```
ヒットチェックの下準備をします。

●HITC_
```
    in          : non
    out         : non
    break       : AF
```
ヒットチェックを行います。

●MODC_
```
    in          : HL = Mode table
    out         : non
    break       : AF,HL
```
ユーザープログラムのモードコントロールをします。使い方はスケルトンを参照。

●SINC_
```
    in          : non
    out         : non
    break       : AF
```
サイン移動に必要な値を計算します。計算した値は，XDF，YDFにセットされます。

●MOVE_
```
    in          : non
    out         : non
    break       : AF
```
SINC_コールで計算した値に従ってキャラクターを移動させます。

●SINM_
```
    in          : non
    out         : non
    break       : AF
```
サイン計算と移動を同時に行います。計算された移動値は残しません。

●DIRS_
```
    in          : H=Target X position
                  L=Target Y position
    out         : A=Target direction
    break       : AF,HL
```
指定の座標の方向を計算します。

●OUTC_
```
    in          : non
    out         : Carry on=screen out
    break       : AF
```
画面外チェックを行ってキャリーに結果を返します。

●OPTM_
```
    in          : H=Target X position
                  L=Target Y position
    out         : non
    break       : AF,HL
```
指定座標を減速追尾します。

個人的な要望または単なるひとり言

今回，僕の意志をついで(?)，相沢氏が制作したシューティングゲームコアシステムをじっくり読ませていただきました。そこで，個人的な要望を少しだけ書かせてもらいます。

まず，ライン描画ルーチンがなくなったのはちょっと残念でした。僕自身は，ライン描画機能を使って，多関節キャラクター，多関節レーザーなどの特殊用途を考えていたんです。

たかがラインといっても使い方しだいでは，かなり表現の幅が広がるでしょう（結構処理が重たいと思いますが，特定のボスキャラに使うなど，制限を設ければなんとかなるはず)。これは，ぜひとも復活を願いたいのですが，どうでしょうか？

あとは，当たり判定部分でひとついわせてもらいましょう。現段階では，完全にキャラクターどうしの判定を座標比較によって行っていますね。それを別ページの仮想画面にあるキャラクタとの判定も行うと便利かな，と思っているわけです。

理由は，当たるとダメージを受ける背景を作りたいからです。もちろん，VRAMAというコールによって，キャラクターがいるVRAMアドレスを求めて判定もできますが，やはりシステムコール一発が楽です。

結構，融通のきかないものになりそうなのでいらないといえばいらないのかもしれませんが，うまくいけば判定処理の高速化もできるかもしれませんからね。

全体的には僕の作ったものよりもすっきりしているし，まとまり方も文句はありません。ぜひ，読者の皆さんの意見を聞きたいところです。

（坂巻克巳）

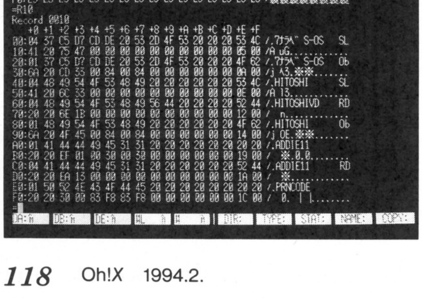
THE SENTINEL

全機種共通
S-OS"SWORD"要

S-OSで学ぶ Z80マシン語講座(3)

Itou Masahiko
伊藤 雅彦

今回から新しいコマンドを追加していきます。まずは，クラスタ情報を表示するFコマンドです。また，11月号で掲載したリストに不備があったため修正するようにしてください。



今回はいきなりお詫びと訂正です。初回に掲載したプログラムのうち，Rコマンド関連の部分を下のコラムに従って修正してください。前のものにバグがあったというわけではないのですが，ほかのコマンドの処理ルーチンとの関連上，修正したほうがいいということになったんです。これもすべて私のいき当たりばったりのプログラミングが災いした設計ミスなわけでして，申しわけなく思っております（でもこのプログラミングスタイルは変えようがない）。

Rコマンドの概要

では，初回掲載のプログラムの中でまだ説明していなかったRコマンド（7～63行）を説明しましょう。Rコマンドはセクタのダンプ表示で，パラメータとして表示したいレコード番号を任意桁の16進数で指定することができます。パラメータ省略時は，前回のRコマンドで表示したセクタの次のセクタを表示します。

また，画面モードが80桁なら1行につき16バイト分のデータを表示しますが，40桁の場合には画面が崩れないように1行につき8バイトだけ表示します。ところが，ここで問題が出てきます。40桁モードのときには1セクタ（256バイト）のデータを表示するのに32行必要になって，表示中に画面がスクロールしてしまい，最初のほうのデータが一瞬にして画面から消え去ってしまうんですね。それではまずいので，最初の16行だけ表示した時点でいったん表示を停止して，なんらかのキーが押されたら残りの16行の表示を再開するようにします。

リストの修正方法

1993年11月号掲載リストのうち，以下の行を修正してください。

・168～187行	削除し，"CALL SCALE"を挿入
・192行	"RCOM3"→"RCOM2"
・194行	"RCOM4"→"RCOM3"
・196行	"RCOM5"→"RCOM4"
・197行	削除
・200行	"RCOM4"→"RCOM3"
・207行	"RCOM3"→"RCOM2"
・209行	"RCOM5"→"RCOM4"
・264行	削除（仕様変更によりAレジスタの引数はなくなったということ）
・270行	削除
・273行	"LD D, A"に変更
・285行	"LD A, D"に変更
・289行	"+1"を削除
・309行の前	"LD A, "/", "CALL #PRINT"を挿入
・410行	"18"→"17"

この処理を実現するには，

1) パラメータの16進数を取得
　→ 16進数でなければエラー処理
　パラメータがなければワークRECORDの値を取得
2) 読み込むセクタのレコード番号を表示
3) セクタの読み込み
　→ 失敗したらエラー処理
4) ワークRECORDの値を，読み込んだセクタのレコード番号+1に更新
5) ダンプを16行だけ表示（画面モードにより表示バイト数を変える）
6) 256バイトすべて表示していなければ，キー入力待ちをしたあとで5）に戻る

という流れにしてやればよさそうです。

PARAMETERルーチン

さて，最初にパラメータの16進数を取得しなければならないのですが，この処理はこれから作るほかのコマンドのパラメータ処理でも必要になってくるものですよね。レコード番号やファイル番号などは16進数で入力されますから。そこで，この部分はひとつのサブルーチンにして，いろいろな処理で使い回せるようにします。

PARAMETERルーチン（196～234行）がそのサブルーチンで，これをコールするとKBPTRの指す位置からパラメータを取り出し，HLにパラメータの値を格納して戻ってきます。また，いつも16進数のパラメータが入力されてあるとは限りませんから，そのときはそれとわかるようにしておかなければいけません。パラメータがなかったときはZフラグが1になり，パラメータはあったけど16進数でなかったときには，Cyフラグが1になってリターンするという仕様にします。

プログラムを説明しましょう。まず，SPCUTルーチンでパラメータの最初の文字を取得します。ここで得られた文字コードが00_Hなら，KBPTRが行の終わりまできてしまった，つまりパラメータがなかったということですから，Zフラグを1にしてリターンします。もっとも，Zフラグをわざわざ1にするという操作は必要ありませんよね。「OR A」でZフラグが1になったかどうかということがリターン条件なんですから。また，Cyフラグには「パラメータはあったけど16進数でなかった」という意味をもたせてありますので，ここは0にしなくてはいけないんですが，それもOR命令では

必ずCyが0になりますからOKです。

そして，ラベルPARAMETER1からPARAMETER2の手前まではループになっています。ここでパラメータの文字を次々に読みながら，HLにパラメータの数値を作っています。

まずループに入る前に，HLを0にしてDEにKBPTRの値を入れています。このときDEにはパラメータの2番目の文字を指すアドレスが入るということになりますね。

そしてループに入るわけですが，ループの先頭で各レジスタの値は，

A ……いまから数値に変換する文字
HL……いままでの文字を変換した数値
DE……Aレジスタに入っている文字の次の文字を指すアドレス

となっていることを押さえてください。

ループの最初に大文字変換をして#HEXをコールしています。このルーチンをコールしてCyフラグが1になって戻ってきたら，パラメータが16進数でなかったということですから，Cyフラグを1にしてリターンします。もっともCyフラグはすでに1になっていますが。ただ，いきなりリターンするとZフラグの状態が保証されませんから，PARAMETER3にジャンプしてZフラグを0にしてからリターンしています。

パラメータが無事に16進数値に変換されたら，HL＝HL×16＋Aという計算をします。HLレジスタにはいままでに変換した数値が入っていますが，まだその下の桁があったということで，まずHLレジスタを16倍してひとつ桁を上げます。$001A_H$なら$01A0_H$というように。HLレジスタを16倍するには，「ADD HL,HL」で2倍になりますから，これを4つ並べればいいわけです。そしてそこに変換したAレジスタの値を足してやれば，HLレジスタが新しい値に更新されます。

そして，「LD A,(DE)」でAレジスタに次の文字を入れています。その文字がスペースか00_Hだったらパラメータの文字を最後まで処理したことになりますから，ループを抜けています。それ以外はループの先頭に戻ります。ループを抜けたら，KBPTRをパラメータの次のアドレスに更新して，Zフラグを0にしてリターンしています。Cyフラグはなにもしなくても0になっていますから，操作する必要はありません。

Rコマンド本体

今度はRコマンド本体（10～63行）を見ていきましょう。ラベルRCOM以降がそれです。先ほど書いた流れを念頭に置いて眺めてみれば，理解するのは難しくないはずです。そろそろプログラムが読めるようになってきたと思いますので，リストをなめるような説明はもういらないでしょう。

セクタの読み込み部分では，#DRDSBルーチンを使ってS-OS内のデータバッファにデータを読み込んでいます。1セクタ分だけなら，S-OSの中に読み込み領域が用意されているんです。

ここでEX命令が使われていますが，これはDEレジスタとHLレジスタの値を交換する命令です。ここでは単にHLレジスタの値をDEレジスタに移すために使っています。まともにLD命令を使おうとすると，

```
LD   E,L
LD   D,H
```

と2命令になって，命令バイト数の面でも実行速度の面でも劣ってしまうんです。

ワークRECORDの更新と一緒にワークWRCBKに1を入れていますが，これはどういうことかよくわからないと思いますので，このへんでWRCBKの意味をちゃんと説明しておきましょう。Rコマンドを使ったあとでWコマンドをパラメータなしで入力したときに，パラメータ省略時値としてワークRECORDの値を取ると，Rコマンドで表示した次のセクタに書き込みをすることになります。でも，セクタ表示をして「あ，このセクタを書き換えたいな」と思ったときにWコマンドをパラメータ省略で使えたほうが便利ですよね。だからこの場合にはRECORDの値そのものじゃなくて，それを－1した値が省略時値になったほうがいいわけです。そこでWRCBKというワークを作って，この値が1のときにはWコマンドの省略時値をRECORDの値－1にする，という意味をもたせているというわけです。わかったでしょうか。細かいことですからわからなくてもいいのですが，プログラムを作ろうとしたらこういう細かい処理も必要になる，あるいはちょっとした処理を加えることで使い勝手がよくなるということは心にとめておいてください。

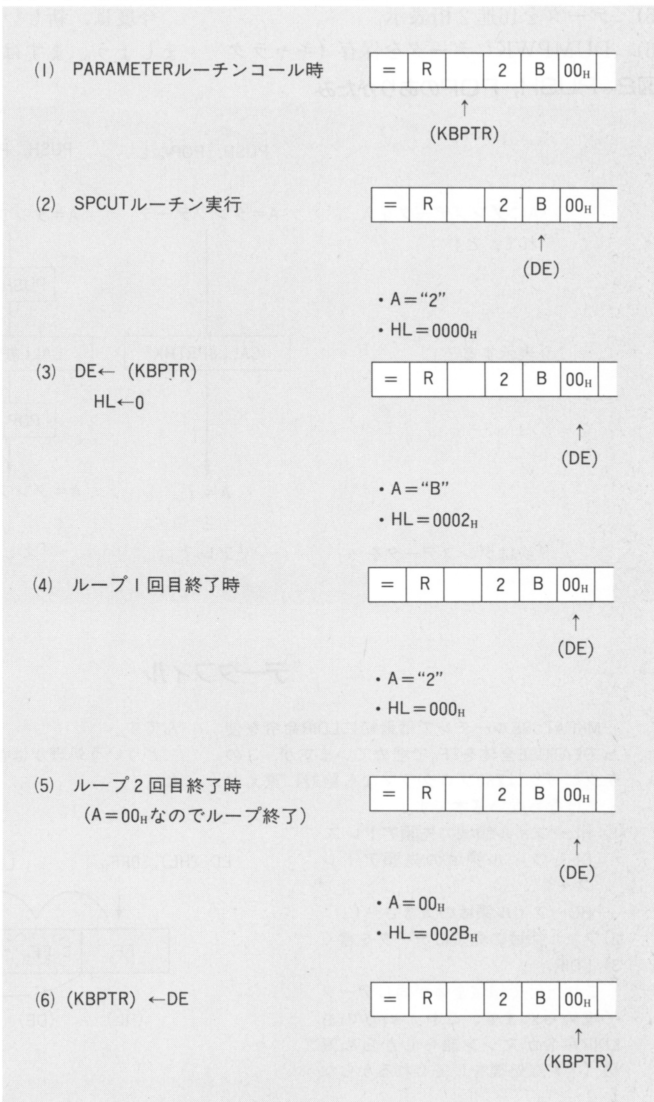
ワーク更新のあと，SCALEルーチン（リスト236行～259行）をコールしています。ここは今回の修正でサブルーチン化したところです。この部分はほかのコマンド処理でも使えるなあって気がついたもので。このルーチンは，ダンプ表示をする前に，

“＋0 ＋1 ＋2 ＋3 ＋4 ＋5 ＋6 ……”

という目盛りを表示するものです。

それからDUMPルーチン（145～195行）がありますが，これは1行分のダンプ表示をするものです。HLの値のアドレスから

図1 PARAMETERルーチンの動作



画面モードにより8バイトか16バイトを1行に表示して，HLの値を次の行で表示すべきデータの先頭アドレスに更新してリターンします。

DUMPはワークとしてDUMPOSをもっています。ここには行の最初に表示する2桁の数字，オフセット値っていうのでしょうか，

```
00：3E 40 CD 95 8F ……
10：80 21 AE 94 CD ……
```

と表示したときの，最初の00とか10とかの数字，あれを入れておくんです。ここに0を入れてDUMPをコールすると，

```
00： ……
```

と表示されるということです。

DUMPルーチンの流れを書いておくと，

1)　DUMPOSの値を16進2桁表示
2)　画面モードによりBレジスタに表示バイト数を設定
3)　DUMPOSを更新
4)　ダンプデータ1バイトを取得
5)　データを16進2桁表示
6)　DUMPWKにデータを保存（キャラクタ表示用）
7)　表示バイト数分のデータを表示していなければ4）へループ
8)　DUMPWKに保存したデータをキャラクタ表示

となります。

命令関係で注目してほしいのはPUSH，POP命令です。これがこの命令の最もオーソドックスな使い方です。#PRTHXはAレジスタを破壊（コール前とは違う値に書き換えられる）する仕様になっていますから，#PRTHXのコールによって，それまでAレジスタに入っていたダンプデータは破壊されてしまいます。しかし，ダンプデータはコール後にも必要になってきますから，破壊されては困ります。こんなとき，PUSH，POPでスタックにデータを一時的に保存しておけばもう安心というわけです（図2）。

Fコマンド

今度は，新しいコマンドを追加していきましょう。まずはFコマンド（64～84行）からです。各クラスタの使用状況をダンプ表示のような形で表示します（図3）。ここからは連載の初回で説明したファイル管理の仕組みを思い出しながら読んでくださいね。

大まかな処理手順を考えてみると，

1)　ディレクトリとFATを読み込む
2)　クラスタの使用状況を表すデータを作成
3)　データを表示

となります。80桁モードのときには画面表示に余裕がありますから，ついでに

4)　FATの生データを表示

というのも加えましょう。

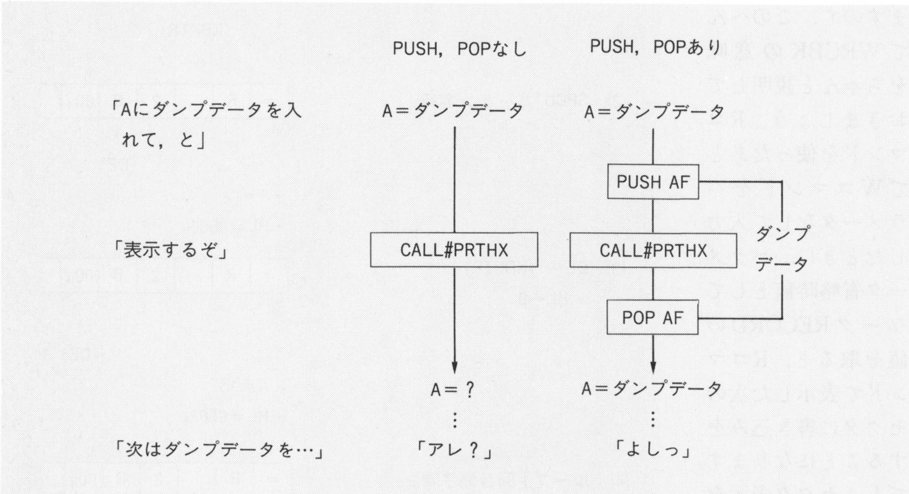
1)のディレクトリ，FAT読み込みのプログラムは難しくないでしょう。FATはS-OS内にFATバッファがありますからそこに読み込んで，ディレクトリはユーザーエリアにバッファ（ラベル名：DIRBF）を用意して読み込みます。また，ディレクトリ領域は16セクタありますが，余計な未使用部分を読み込まないように1セクタずつ読むことにします。実際に作ったのがサブルーチンREADDIR（261～298行），READFAT（299～315行）ですので，コメントを頼りにして解読してください。

3）のデータ表示はRコマンドのダンプ表示と似たような処理ですから楽にできそうです。FCOMS1（85～139行）がそのルーチンで，4)のFAT表示処理もこのルーチンを使っています。Rコマンドの処理と最も違うのは，表示データがFD_H，FE_H，FF_Hだったらそれぞれ“==”，“＊＊”，“..”と表示する点でしょうか。

ややこしそうなのが2)のクラスタ使用状況を調べる処理です。具体的にはFATUSEという128バイトのワークエリアを用意して，そこに第0クラスタから第127クラスタまでの使用状況を表すデータをセットする，という処理です。セットするデータは，

01_H～$7E_H$：使用ファイル番号
FD_H　：使用不可クラスタ
FE_H　：複数のファイルで重複使用

図2　PUSH，POPのありがたみ



データフィル

MKFATUSEルーチンでは最初にLDIR命令を使ってFATUSE全体をFF_Hで埋めていますが，このテクニックはZ80プログラマなら絶対に覚えなくてはならない基本です。

1)　HL←フィル領域の先頭アドレス
　DE←フィル領域の先頭アドレス＋1
　BC←フィル領域の大きさ－1
2)　フィル領域の先頭にデータを書く
3)　LDIR

これだけで領域全体が同じデータで埋められます。これというのもLDIR命令がマシン語らしからぬ高級（？）な処理をしてくれるからなんです。

どういう処理かは参考書で確認してくださいね。

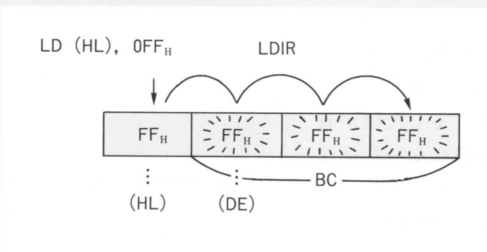


図3　Fコマンドの表示形式

```
     +0 +1 +2 +3 +4 +5 +6 +7
00 : == == 01 01 02 02 02 04
08 : 04 .. .. 03 ** ** .. ..
              :
78 : == == == == == == == ==

01H～7EH : 使用ファイル番号
==       : 使用不可
..       : 未使用
**       : 複数のファイルで重複使用
           （普通はない）
```

FF_H　　：未使用クラスタ

とします。ファイル番号というのは，ディレクトリに登録されている順にファイルに付けた番号で（KILLされたファイルにも番号が割り当てられる），このディスクエディタではファイルをファイル番号で扱います。

どうやったらこの処理ができるかなと考えてみると，

1)　FATUSEをFF_Hでクリア

2)　ディレクトリに登録されているファイルについて，FATを見て使用ファイル番号および重複使用記号（FE_H）をセット

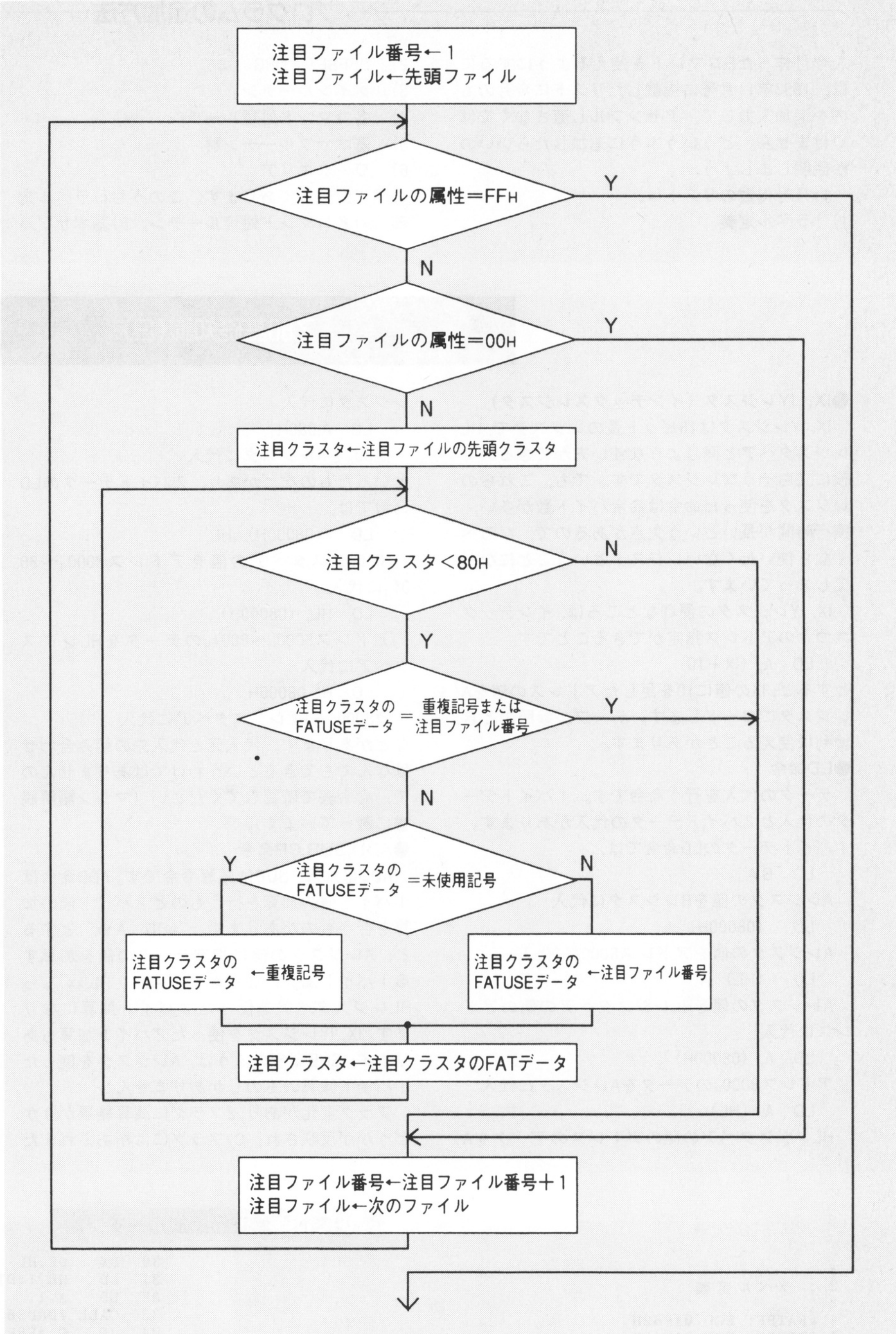
3)　FAT上では未使用クラスタではないのにFATUSEで未使用となっているところに，使用不可記号（FD_H）をセット

という順にやるのがよさそうだということになりました。これをさらに細かくあれこれ考えてできたのが，MKFATUSEというサブルーチンです（316〜390行）。2)の処理をやっている部分は難解かなという感じがしますので，フローチャートを示しておきましょう（図4）。ここではクラスタの重複使用チェック，ループチェック（クラスタのつながりがループしているか）が入っていますが，FAT情報が破壊されていない限りこのチェックに引っかかることはありません。あまり意味のない処理を入れてわかりにくいプログラムにしてしまったのは申しわけないんですが，ディスクエディタは破壊されたディスクの修復に使われることもありますから，こういうところもしっかり作っておく必要があるんじゃないかと思うんです。

＊　　＊　　＊

今月はここまでです。リストを読むのにも慣れてきたでしょうか。いろんなプログラムを読んで，こういう処理のときはこういうプログラムを組むんだっていうパターンがわかるようになると，自分でプログラムを作るときにも役に立ちますよ。

図4　MKFATUSEのメイン部分



今月使用したシステムサブルーチン&ワークエリア

●サブルーチン

#PRNTS：スペースを表示

#MSX：DEレジスタペアの示すアドレスから00Hがあるまで文字列表示

#PAUSE：スペースが押されていれば，再びなにかキーを押すまでリターンしない。このときSHIFT＋BREAKを押すと，CALL命令の次の2バイトに書かれたアドレスへジャンプ

#BELL：ビープ音を鳴らす

#PRTHX：Aレジスタの値を16進2桁で表示

#PRTHL：HLレジスタペアの値を16進4桁で表示

#ASC：Aレジスタの下位4ビットの値を16進数の文字に変換

例）0BH → "B"（42_H）

#HEX：Aレジスタの16進数の文字を数値に変換。16進数の文字でなかった場合はCy=1

例）"9"（39H） → 09_H

#DRDSB：セクタの読み込み。（#DSK）にデバイス，DEレジスタペアに読み込み開始レコード番号，Aレジスタに読み込みセクタ数，HLレジスタペアに読み込み先アドレスをセットしてコールする

#FLGET：カーソル点滅1文字入力を行い，入力文字をAレジスタにセット

#ERROR：Aレジスタにエラー番号をセットしてコールすると，エラーメッセージを表示してくれる

●ワークエリア

#DTBUF：データバッファの先頭アドレス

#FATBF：FATバッファの先頭アドレス

#DIRPS：ディレクトリが記録されているセクタのレコード番号

#FATPOS：FATが記録されているセクタのレコード番号

#DSK：アクセスしようとするデバイス名

プログラムの追加方法

今月作ったFコマンドを使えるようにするには，1993年11月号に掲載したリストに今月のものを追加入力して，アセンブルし直さなくてはいけません。どういうふうに追加したらいいのか説明しましょう。

11月号掲載のリストは，

1) ラベル定義
2) OFFSET，ORG
3) メインルーチン
4) 各コマンド処理ルーチン
5) 基本サブルーチン群
6) ワークエリア

の順に記述してあります。このうち1)ラベル定義，4)各コマンド処理ルーチン，5)基本サブルーチン群，6)ワークエリアのそれぞれの最後に，今月分のリストを追加すればいいのです(FCOMS1，DCOMS1は基本サブルーチンではなくコマンド処理ルーチンの一部とみなしてください)。ただし，コマンド処理ルーチンは最後に追加するのではなく，RET命令が書いてあったところに追加してください。

Z80基礎知識(2)

●IX，IYレジスタ（インデックスレジスタ）

IX，IYレジスタは16ビット長のレジスタで，HLレジスタペアと同じような使い方ができるので役に立ちそうなレジスタです。でも，これらのレジスタを使った命令は命令バイト数が多い，実行時間が長いという欠点があるので，なるべくなら使いたくないレジスタということになってしまっています。

IX，IYレジスタの便利なところは，インデックスつきのアドレス指定ができることです。

LD A,(IX＋10)

とすると，IXの値に10を足したアドレスの値をAレジスタにロードします。テーブル参照のとき，便利に使えることがあります。

●LD命令

データの代入を行う命令です。1バイトデータの代入と2バイトデータの代入があります。1バイトデータのLD命令では，

LD B,A

Aレジスタの値をBレジスタに代入

LD (08000H),A

Aレジスタの値をアドレス8000_Hに代入

LD (HL),A

Aレジスタの値をHLレジスタペアの値のアドレスに代入

LD A,(08000H)

アドレス8000_HのデータをAレジスタに代入

LD A,(HL)

HLレジスタペアの値のアドレスのデータをAレジスタに代入

LD A,080H

値80_HをAレジスタに代入

といったものなどがあり，2バイトデータのLD命令では，

LD (08000H),HL

HLレジスタペアの値をアドレス8000_H〜8001_Hに代入

LD HL,(08000H)

アドレス8000_H〜8001_HのデータをHLレジスタペアに代入

LD HL,08000H

値8000_HをHLレジスタペアに代入

などがあります。代入元と代入先の組み合わせはなんでもできるというわけではありませんので，命令表で確認してください（マシン語解説書に載っています）。

●ADD,SUB,CP命令

ADDは加算，SUBは減算命令です。ADD命令は1バイト値の加算を行うものと2バイト値の加算を行うものがあります。“ADD A,xx”とすると，Aレジスタの値に指定データの値を加算する1バイト加算になります。“ADD HL,xx”ならHLレジスタペアを使った2バイト加算になります。IX，IYレジスタを使った2バイト加算もあります。SUB命令のほうは，Aレジスタを使った1バイト減算のものしかありません。

フラグ変化があり，Zフラグに演算結果が0かどうかが反映され，Cyフラグには桁あふれまたは桁借りが起きたかどうかが反映されます。

CP命令はSUB命令と同様の減算処理をしますが，その結果をAレジスタに入れないで捨ててしまいます。つまりAレジスタの値は命令実行前と変わりません。変化するのはフラグだけです。フラグ変化により2つの値の大小比較ができます。たとえば，“CP B”を実行して，

Cy=1ならばA<B

Z=1ならばA=B

Cy=0,Z=0ならばA>B

と判断できます。

●AND,OR,XOR命令

論理積，論理和，排他的論理和の論理演算を行います。Aレジスタと指定データとの間で論理演算を行い，結果をAレジスタに入れます。Aレジスタの任意のビットをリセット，セット，反転する命令と考えることもできます。

命令実行後にフラグ変化があって，演算結果が0ならZフラグが1になり，それ以外なら0になります。Cyフラグは必ず0になります。

●INC,DEC命令

レジスタなどの値のインクリメント（＋1すること），デクリメント（－1すること）をする命令です。至って単純な命令ですが，気をつけなくてはいけないのがフラグ変化です。演算命令なのに，桁あふれや桁借りが起きてもCyフラグが変化しません。それからHLレジスタペアなど，2バイト値のインクリメント，デクリメントはZフラグの変化もないので注意が必要です。

リスト 修正&追加ルーチン

```
;
; ラベル定義
;
#FATBF: EQU 01F62H
#DIRPS: EQU 01F60H
#FATPOS:EQU 01F5EH
;
; コマンド処理ルーチン
;
;
; R Command
;
RCOM:
 CALL PARAMETER
 JP   C,#BELL
;
 JR   NZ,RCOM1
 LD   HL,(RECORD)
;
RCOM1:
 CALL PRTRSET
 CALL #MPRNT
 DM   'Record '
 DB   0
 CALL #PRTHL
 CALL #LTNL
;
 LD   A,(DEVICE)
 LD   (#DSK),A
 EX   DE,HL
 LD   HL,(#DTBUF)
 LD   A,1
 CALL #DRDSB    ; セクタ読み込み
 JP   C,#ERROR
;
 INC  DE
 LD   (RECORD),DE
 LD   A,1
 LD   (WRCBK),A
;
 CALL SCALE
;
 XOR  A         ; A=0
 LD   (DUMPOS),A
 LD   HL,(#DTBUF)
RCOM2:
 LD   C,16
RCOM3:
 CALL #PAUSE
 DW   RCOM4
 CALL DUMP      ; セクタデータ1行表示
 DEC  C
 JR   NZ,RCOM3
;
 LD   A,(DUMPOS)
 OR   A
 RET  Z
;
```


▶さめたポテトをトースターであっためていると，ポテトがトースターの中でメラメラと燃え上がっていた。あわててトースターごと外に持ち出したら，周囲の人に変な目で見られた。
中島 貴史(17)滋賀県

```
 CALL #FLGET
 JR   RCOM2
;
RCOM4:
 RET
;
; F Command
;
FCOM:
 CALL READDIR
 CALL NC,READFAT
 RET  C           ; READDIR または READFAT でエラー
 CALL MKFATUSE
;
 CALL PRTRSET
 CALL SCALE
 LD   HL,FATUSE
 CALL FCOMS1      ; FATUSE 表示
 RET  C
 LD   A,(WIDMODE)
 OR   A
 RET  Z           ; 40桁モードなら終了
;
 CALL #LTNL
 LD   HL,(#FATBF)
;JR   FCOMS1      ; FAT表示
;
;  in ---- HL = 表示データ先頭アドレス
;  out --- Cy = ブレイクキーが押された(1)／押されない(0)
;  break - F, A, BC, D, HL
;
FCOMS1:
 LD   C,0         ; C = オフセット表示値
FCOMS11:
 CALL #PAUSE
 DW   FCOMS16     ; SHIFT+BREAK なら FCOMS16 へ
;
 LD   A,C
 CALL #PRTHX
 LD   A,':'
 CALL #PRINT
;
 LD   A,(WIDMODE)
 OR   A
 LD   B,8
 JR   Z,FCOMS12
 LD   B,16        ; B = 表示バイト数
;
FCOMS12:
 LD   A,(HL)
 INC  HL
 INC  C
 CP   0FDH
 JR   C,FCOMS14 ; FCH 以下なら普通に表示
 LD   D,'.'
 INC  A           ; FFH（未使用）か
 JR   Z,FCOMS13
 LD   D,'*'
 INC  A           ; FEH（重複）か
 JR   Z,FCOMS13
 LD   D,'='
FCOMS13:
 LD   A,D
 CALL #PRINT
 CALL #PRINT
 JR   FCOMS15
FCOMS14:
 CALL #PRTHX
FCOMS15:
 CALL #PRNTS
 DJNZ FCOMS12
;
 CALL #LTNL
 LD   A,C
 CP   080H
 JR   NZ,FCOMS11
 RET              ; （この時必ず Cy=0 になっている）
;
FCOMS16:
 SCF
 RET
;
; 基本サブルーチン群
;
;==================== Rコマンド分
;================= 既掲載分
;
; DUMP
;
;  in ---- HL = 表示開始アドレス
;  out --- HL = 表示最終アドレス＋1
;  break - F, A, B, DE
;
DUMP:
 LD   A,(DUMPOS)
 LD   D,A
 CALL #PRTHX
 LD   A,':'
 CALL #PRINT
;
 LD   A,(WIDMODE)
 OR   A
 LD   B,8
 JR   Z,DUMP1
 LD   B,16        ; B = 表示バイト数
;
DUMP1:
 LD   A,D
 ADD  A,B
 LD   (DUMPOS),A
;
 LD   DE,DUMPWK
DUMP2:
 LD   A,(HL)
 INC  HL
 PUSH AF          ; ダンプデータ保存
 CALL #PRTHX
 CALL #PRNTS
 POP  AF
;
 CP   020H
 JR   NC,DUMP3
 LD   A,'.'
DUMP3:
 LD   (DE),A
;
 INC  DE
 DJNZ DUMP2
;
 XOR  A           ; A=0
 LD   (DE),A
 LD   A,'/'
 CALL #PRINT
 LD   DE,DUMPWK
 CALL #MSX
 CALL #PRNTS      ; 全角文字に対応
 JP   #LTNL
;
; PARAMETER
;
;  out --- Z  = パラメータがない(1)／ある(0)
;          Cy = パラメータが16進数として無効(1)／有効(0)
;          HL = パラメータの値（Z=0 かつ Cy=0 の時）
;  break - F, A, DE, HL
;
PARAMETER:
 CALL SPCUT
 OR   A
 RET  Z           ; パラメータがない
;
 LD   HL,0
 LD   DE,(KBPTR)
PARAMETER1:
 CALL CAPITAL
 CALL #HEX
 JR   C,PARAMETER3 ; パラメータが16進数として無効
 ADD  HL,HL
 ADD  HL,HL
 ADD  HL,HL
 ADD  HL,HL       ; 16倍
 ADD  A,L
 LD   L,A
 LD   A,(DE)
 OR   A
 JR   Z,PARAMETER2
 INC  DE
 CP   ' '
 JR   NZ,PARAMETER1
;
PARAMETER2:
 LD   (KBPTR),DE
;
PARAMETER3:
 LD   A,0
 INC  A           ; Cy を変えずに Z=0 にする
 RET
;================= 今回追加分
;
; SCALE
;
;  break - F, A, B, DE, H
;
SCALE:
 CALL #PRNTS
 CALL #PRNTS
 LD   H,0         ; H = 目盛りの数値
 LD   B,16
 LD   A,(WIDMODE)
 OR   A
 JR   NZ,SCALE1
 LD   B,8         ; B = 目盛りの長さ
SCALE1:
 CALL #MPRNT
 DM   ' +'
 DB   0
 LD   A,H
 CALL #ASC
 CALL #PRINT
 INC  H
 DJNZ SCALE1
 JP   #LTNL
;===================== Fコマンド分
;
; READDIR
```

▶京都の冬は寒い。コタツしかないのだが，無事に冬を越せるのか？
奥山　貴士(21)京都府

```
;
;   out --- Cy = 読み込み失敗(1)/成功(0)
;   break - F, A, BC, DE, HL, F',A'
;
READDIR:
 LD   DE,(#DIRPS)
 LD   HL,DIRBF
READDIR1:
 LD   A,(DEVICE)
 LD   (#DSK),A
 LD   A,1
 CALL #DRDSB     ; ディレクトリ1セクタ読み込み
 JR   C,READDIR2 ; 読み込みエラー
 INC  DE
 LD   BC,000E0H
 ADD  HL,BC      ; (ここで必ず Cy=0 になる)
 LD   A,(HL)
 INC  A          ; A = FFH なら Z=1
 RET  Z          ; 読み込み正常終了
 LD   BC,00020H
 ADD  HL,BC
 LD   A,DIRBF/256+15
 CP   H          ; 16セクタ読み込んだら Cy=1
 JR   NC,READDIR1
;
 CALL #MPRNT
 DM   'Bad Directory Data'
 DB   00DH,0
 CALL #BELL
 SCF
 RET
;
READDIR2:
 CALL #ERROR
 SCF
 RET
;
; READFAT
;
;   out --- Cy = 読み込み失敗(1)/成功(0)
;   break - F, A, BC (Cy=1 の時), DE, HL, F',A'
;
READFAT:
 LD   A,(DEVICE)
 LD   (#DSK),A
 LD   DE,(#FATPOS)
 LD   HL,(#FATBF)
 LD   A,1
 CALL #DRDSB
 RET  NC
 CALL #ERROR
 SCF
 RET
;
; MKFATUSE
;
;   break - F, A, BC, DE, HL, IX
;
MKFATUSE:
 LD   HL,FATUSE
 LD   DE,FATUSE+1
 LD   BC,127
 LD   (HL),0FFH
 LDIR            ; FATUSE を FFH で埋める
;
 LD   B,1
 LD   IX,DIRBF
 JR   MKFATUSE7
MKFATUSE1:
 DEC  A          ; 属性が 00H か
 JR   Z,MKFATUSE6
 LD   E,(IX+30)  ; E = 先頭クラスタ
 LD   D,0
 JR   MKFATUSE5
;
MKFATUSE2:
 LD   HL,FATUSE
 ADD  HL,DE
 LD   A,(HL)
;
 INC  A          ; FFH (未使用) か
 JR   NZ,MKFATUSE3
 LD   (HL),B     ; FFH ならファイル番号を入れる
 JR   MKFATUSE4
MKFATUSE3:
 INC  A          ; FEH (重複) か
 JR   Z,MKFATUSE6
 SUB  2
 CP   B          ; クラスタ連鎖がループしているか
 JR   Z,MKFATUSE6
 LD   (HL),0FEH  ; FEH を入れる
;
MKFATUSE4:
 LD   HL,(#FATBF)
 ADD  HL,DE
 LD   E,(HL)
MKFATUSE5:
 LD   A,E
 CP   080H       ; クラスタ連鎖終端か
 JR   C,MKFATUSE2
;
MKFATUSE6:
 LD   DE,32
 ADD  IX,DE      ; 次のファイルへ
 INC  B
MKFATUSE7:
 LD   A,(IX+0)   ; A = 属性
 INC  A          ; FFH か
 JR   NZ,MKFATUSE1
;
 LD   DE,(#FATBF)
 LD   HL,FATUSE
 LD   B,128
MKFATUSE8:
 LD   A,(DE)
 OR   A
 JR   Z,MKFATUSE9 ; クラスタ未使用ならスキップ
 LD   A,(HL)
 INC  A
 JR   NZ,MKFATUSE9 ; クラスタ使用状態が
                   ; 設定されていればスキップ
 LD   (HL),0FDH
MKFATUSE9:
 INC  DE
 INC  HL
 DJNZ MKFATUSE8
 OR   A          ; Cy=0
 RET
;
; ワークエリア
;
FATUSE:          ; FATの使用状況
 DS 128
DIRBF:           ; ディレクトリデータ操作領域
 DS 256*16
```

▶ 全機種共通システムインデックス ◀

＊以下のアプリケーションは，基本システムであるS-OS "MACE" またはS-OS "SWORD" がないと動作しませんのでご注意ください。

▶「餓狼伝説２」が出るんだから，「餓狼伝説SPECIAL」も出してほしいと思うんですけど……。
荒明　慎久(14)北海道

1986
第22部　magiFORTH TRACER
第23部　ディスクダンプ＆エディタ
第24部　“SWORD”2000 QD
連載　対話で学ぶmagiFORTH
特別付録　PC-8801版S-OS“SWORD”
■86年7月号
第25部　FM音源ミュージックシステム
付録　FM音源ボードの製作
連載　計算力アップのmagiFORTH
特別付録　SMC-777版S-OS“SWORD”
■86年8月号
第26部　対局五目並べ
第27部　MZ-2500版S-OS“SWORD”
■86年9月号
第28部　FuzzyBASIC発表
連載　明日に向かってmagiFORTH
■86年10月号
第29部　ちょっと便利な拡張プログラム
第30部　ディスクモニタDREAM
第31部　FuzzyBASIC料理法<1>
■86年11月号
第32部　パズルゲームHOTTAN
第33部　MAZE in MAZE
連載　FuzzyBASIC料理法<2>
■86年12月号
第34部　CASL & COMET
連載　FuzzyBASIC料理法<3>

1987
■87年1月号
第35部　マシン語入力ツールMACINTO-C
連載　FuzzyBASIC料理法<4>
■87年2月号
第36部　アドベンチャーゲームMARMALADE
第37部　テキアベ作成ツールCONTEX
■87年3月号
第38部　魔法使いはアニメがお好き
第39部　アニメーションツールMAGE
付録　“SWORD”再掲載とMAGICの標準化
■87年4月号
第40部　INVADER GAME
第41部　TANGERINE
■87年5月号
第42部　S-OS“SWORD”変身セット
第43部　MZ-700用“SWORD”をQD対応に
■87年6月号
インタラプト　コンパイラ物語
第44部　FuzzyBASICコンパイラ
第45部　エディタアセンブラZEDA-3
■87年7月号
第46部　STORY MASTER
■87年8月号
第47部　パズルゲーム碁石拾い
第48部　漢字出力パッケージJACKWRITE
特別付録　FM-7/77版S-OS“SWORD”
■87年9月号
第49部　リロケータブル逆アセンブラInside-R
特別付録　PC-8001/8801版S-OS“SWORD”
■87年10月号
第50部　tiny CORE WARS
第51部　FuzzyBASICコンパイラの拡張
第52部　X1turbo版S-OS“SWORD”
■87年11月号
序論　神話のなかのマイクロコンピュータ
付録　S-OSの仲間たち
第53部　もうひとつのFuzzyBASIC入門
第54部　ファイルアロケータ＆ローダ
インタラプト　S-OSこちら集中治療室
第55部　BACK GAMMON
■87年12月号
第56部　タートルグラフィックパッケージTURTLE
第57部　X1turbo版“SWORD”アフターケア
ラインプリントルーチン
特別付録　PASOPIA7版S-OS“SWORD”

1988
■88年1月号
第58部　FuzzyBASICコンパイラ・奥村版
付録　石上版コンパイラ拡張部の修正
■88年2月号
第59部　シューティングゲームELFES
■88年3月号
第60部　構造型コンパイラ言語SLANG
■88年4月号
第61部　デバッギングツールTRADE
第62部　シミュレーションウォーゲームWALRUS
■88年5月号
第63部　シューティングゲームELFESⅡ
第64部　地底最大の作戦
■88年6月号
第65部　構造化言語SLANG入門(1)
第66部　Lisp-85用NAMPAシミュレーション
■88年7月号
第67部　マルチウィンドウドライバMW-1
連載　構造化言語SLANG入門(2)
■88年8月号
第68部　マルチウィンドウエディタWINER
■88年9月号
第69部　超小型エディタTED-750
第70部　アフターケアWINERの拡張
■88年10月号
第71部　SLANG用ファイル入出力ライブラリ
第72部　シューティングゲームMANKAI
■88年11月号
第73部　シューティングゲームELFESⅣ
■88年12月号
第74部　ソースジェネレータSOURCERY

1989
■89年1月号
第75部　パズルゲームLAST ONE
第76部　ブロックゲームFLICK
■89年2月号
第77部　高速エディタアセンブラREDA
特別付録　X1版S-OS“SWORD”<再掲載>
■89年3月号
第78部　Z80用浮動小数点演算パッケージSOR
OBAN
■89年4月号
第79部　SLANG用実数演算ライブラリ
■89年5月号
第80部　ソースジェネレータRING
■89年6月号
第81部　超小型コンパイラTTC
■89年7月号
第82部　TTC用パズルゲームTICBAN
■89年8月号
第83部　CP/M用ファイルコンバータ
■89年9月号
第84部　生物進化シミュレーションBUGS
■89年10月号
第85部　小型インタプリタ言語TTI
■89年11月号
第86部　TTI用パズルゲームPUSH BON!
■89年12月号
第87部　SLANG用リダイレクションライブラリDIO.LIB

1990
■90年1月号
第88部　SLANG用ゲームWORM KUN
特別付録　再掲載SLANGコンパイラ
■90年2月号
第89部　超小型コンパイラTTC++
■90年3月号
第90部　超多機能アセンブラOHM-Z80
■90年4月号
第91部　ファジィコンピュータシミュレーションI-MY
■90年5月号
第92部　インタプリタ言語STACK
■90年6月号
第93部　リロケータブルフォーマットの取り決め
第94部　STACK用ゲームSQUASH!
第95部　X68000対応S-OS“SWORD”
特別付録　PC-286対応S-OS“SWORD”
■90年7月号
第96部　リロケータブルアセンブラWZD
■90年8月号
第97部　リンカWLK
■90年9月号
第98部　BILLIARDS
■90年10月号
第99部　ライブラリアンWLB
■90年11月号
第100部　タブコード対応エディタEDC-T
■90年12月号
第101部　STACKコンパイラ

1991
■91年1月号
第102部　ブロックアクションゲームCOLUMNS
■91年2月号
第103部　ダイスゲームKISMET
■91年3月号
第104部　アクションゲームMUD BALLIN'
■91年4月号
第105部　SLANG用カードゲームDOBON
■91年5月号
第106部　実数型コンパイラ言語REAL
■91年6月号
第107部　Small-C処理系の移植
■91年7月号
第108部　REALソースリスト編
■91年8月号
第109部　Small-Cライブラリの移植
■91年9月号
第110部　SLANG用NEWファイル出力ライブラリ
■91年10月号
第111部　Small-C活用講座（初級編）
■91年11月号
第112部　Small-C活用講座（応用編）
第113部　MORTAL
■91年12月号
第114部　Small-C SLANGコンパチ関数

1992
■92年1月号
第115部　LINER
■92年2月号
第116部　シミュレーションゲームPOLANYI
■92年3月号
第117部　カードゲームKLONDIKE
■92年4月号
第118部　オプティマイザO80実践Small-C講座(1)
■92年5月号
第119部　COMMAND.OBJ実践Small-C講座(2)
■92年6月号
第120部　COMMAND.OBJ2実践Small-C講座(3)
■92年7月号
第121部　関数リファレンス実践Small-C講座(4)
■92年8月号
第122部　ワイルドカード実践Small-C講座(5)
第123部　グラフィックライブラリ GRAPH.LIB
■92年9月号
第124部　O-EDIT&MODCNV
■92年10月号
第125部　SLENDER HUL実践Small-C講座(6)
■92年11月号
第126部　EDIT実践Small-C講座(7)
■92年12月号
第127部　MAKE実践Small-C講座(8)

1993
■93年1月号
第128部　EDC-Tの拡張
■93年2月号
第129部　BLACK JACK
■93年3月号
第130部　シューティングゲームコアシステム作成法(1)
■93年4月号
第131部　シューティングゲームコアシステム作成法(2)
■93年5月号
第132部　シューティングゲームコアシステム作成法(3)
■93年6月号
第133部　REVERSI
■93年7月号
特別付録　MSX用S-OS“SWORD”
■93年8月号
第134部　MACINTO-C再掲載
■93年9月号
第135部　7並べ
特別付録　SLANG再々掲載
■93年10月号
第136部　シューティングゲームコアシステム作成法(4)
■93年11月号
第137部　S-OSで学ぶZ80マシン語講座(1)
■93年12月号
第138部　エディタアセンブラREDA再掲載

▶うまくいったら，あと2人X68000ユーザーが増えるかもしれない。みんなで幸せになれるといいなあ……ふう。
岩瀬　貴代美(22)福岡県

第88回

猫とコンピュータ

便利が3日つづくまで

Takazawa Kyoko
高沢 恭子

新しい発想やユニークな視点の新製品やアイデア商品って，とても便利そうです。話題や流行になると，一度は使ってみたくなり，新しもの好きならずともつい手を伸ばしてしまうものですが……。

強かった南風もおだやかになった。黄葉の舞う赤レンガの校門を入ると，広い石畳の道がまっすぐにのびている。右手は図書館と室内プール，左手に校舎。

午後3時半の日ざしを受けて，向こうから下校する生徒の一群がやってくる。最前列で，見おぼえのあるホワイトベージュのハーフコートが日に映えている。女の子もまじえた友人たちと，談笑しながら歩いてくるのはトオルだった。

あたたかな初冬，あと数カ月で卒業というときの，ほんの1コマの光景にすぎないけれど，満ち足りた高校生活を真正面からながめたような数秒だった。

ところで，午後4時と指定された「三者面談」を忘れて帰るつもりなのかしら。立ち止まっていると，トオルは私のところまできて友人たちを紹介し，「じゃあね」と彼らに別れをつげた。

「ちょっと早いんで遊んでたんだ」。担任の先生のいる4階の国語科職員室へ私を案内しながらいった。大学受験の本番をひかえて，最終的な方針を話しあうための，先生と親子による三者面談である。

2年生のはじめから私立文系のある学部を目標ときめ，以来迷うことなくここまでやってきた。学校としては，公立の高校でもあり，進路指導の先生をはじめ，国立の受験をすすめる傾向がある。国立，私立を併願する人も多いが，志望大学がはっきりしているのだから，あまり無用なことはしたくないと彼は考えている。

さいわい担任の加藤先生は，学部はちがうものの，トオルが志望している大学の出身だそうだ。国立の受験にはあまりこだわらず，いろいろとアドバイスをしてくださった。「いいセンをいくと思うんだけど，試験は水モノだからね」と先生。

ともかくやるしかない。トオルも受験の緊張感は楽しいのだそうだ。どんな結果を手にするかわからないが，これも貴重な思い出づくりになることだろう。

ハサミの使いかた

新宿のおばあちゃんほど，ものごとに厳しく，堅実な考えと判断力を持った人はいないだろうと，まわりも自分も思っているらしい。

そのおばあちゃんが，通信販売で「高枝切りバサミ」を買った。テレビでも宣伝している，柄の部分が数メートルにものびるハサミで，庭木の高いところの枝や葉をハシゴを使わずに地上にいたまま切ることができるというものだ。先端の刃の部分は，ノコギリに交換することもできる。

これはムダな買物ではないと思ったのだろう。高い木の枝を切るには，切ることばかりがむずかしいのではない。ひとりでやろうとしたら，まず地上からながめて，あの枝とこの枝を，あのあたりから切ろうと決める。それから自分でハシゴか脚立をたてて登っていくのだが，上までいくと，ハテ，この枝でよかったのか，この辺から切り落としてほんとにいいのか，わかりにくくなるものだ。

家族がたくさんそろっていたころは，父や兄や，私までがハシゴに登らされて，下にいる母の命令であちこちの枝を整理したこともあった。いま弟夫妻との3人暮らしになったおばあちゃんには，人手を借りずにできることがいちばんだ。地上にいて木の姿全体を見わたしながら枝葉をととのえる。これは意匠デザインの方法としても望ましい。高いところでもどんどん登るおばあちゃんだが，こんな便利なものがあるなら，もうあぶないマネもしなくてすむと思ったのも無理はない。

ところが，じっさいに手元にとどいたそのハサミをかかえて，期待いっぱいのおばあちゃんが庭木の下までいったとき，大きな失望が待っていた。

夾竹桃か八重桜か忘れたが，庭の西南の角にある塀ぎわの木だった。おばあちゃんは，一瞬のうちにテレビ実演の盲点に気づいたのだ。問題は木のまわりのスペースである。目ざす枝にハサミをさしのべるとき，「高枝切りバサミ」の長い柄は一直線で，曲げることはできない。そのため木の真下にいたのでは，ほかの枝にじゃまされてハサミが使えない。木のそとに出て，周囲から腕をのばさなくてはダメなのだ。

この木の手前には池があった。すぐ近くには別の木もある。しかも塀ぎわの木となると，しごとの半分は塀の外にまわらなくてはならない。隣は「トヨタ東京オート」の駐車場である。

テレビではじゅうぶんなスペースのある条件のみでの実演だったが，どこでもそうとはかぎらない。たとえ広い庭でも，樹木が植えられる条件はさまざまだろう。木のあるところ，かならずこの新案のハサミが役にたつわけではないということを，さすがのおばあちゃんもハサミを持ってから気づいたのだ。

圧縮と解凍の原理

人の買い物については，あんなものを買ってどうするのだろうと考えるくせに，自分は自分で，たくさんの思惑ちがいや期待はずれの無効な買物をしている。

近くのスーパーで，「アイデア商品」というもののひとつを買った。自宅で散髪するときに，切った髪を床に散らさなくてすむ

ように，肩のまわりのミゾにあつめるポリエステル製のケープで，おもにこどものためのものだ。

床屋さんが嫌いなトオルが理髪店にいったのは，小学校２年生のときの１回だけ。彼の床屋さんはずっと私で，毎回あと始末に時間をかけている。これがあれば掃除はかんたん,18年間の不便が1,680円で解消できると思った。

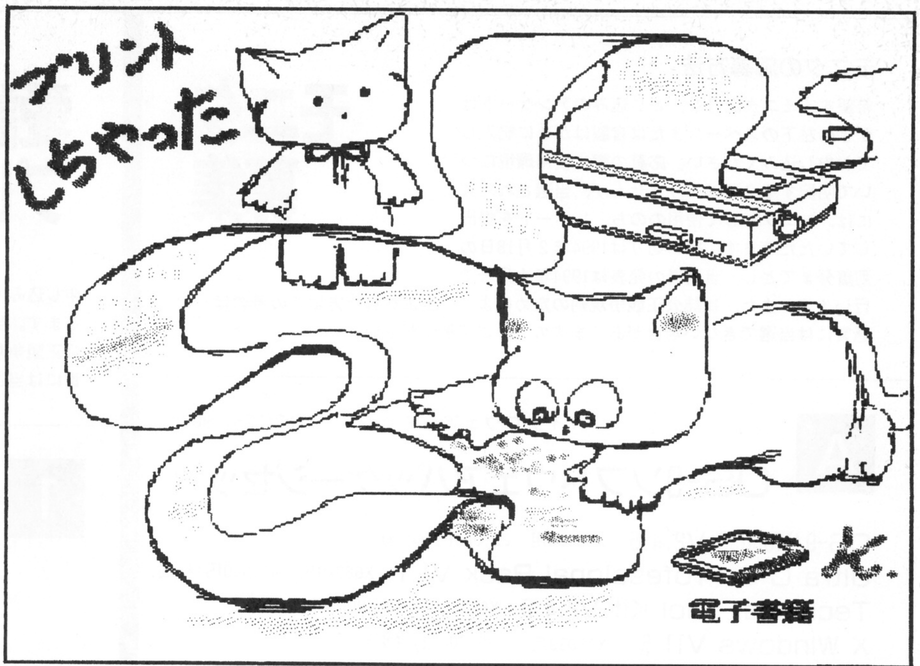
さっそく使ってみた。多くの髪が肩のまわりにできたハンモックのような谷間に入るのは確かだった。だが，散髪のあとは肩からはずした直径60センチの，周囲にワイヤーの入ったタライのような入れ物から，髪の毛を取り出さなくてはいけない。ポリエステルタフタという布はクシャクシャで掃除機もあてにくい。こんなことをするより，床に新聞紙を敷いて布の肩かけをし，切った髪はしぜんに下にすべらせるほうがずっと気持ちのよいあと始末ができる。２度ほど使ったあと，けっきょく元の方法にもどった。

「フトン圧縮袋」の場合はなにもしないで終わった。はじめは，フトンをあんなにペシャンコにしてしまってよいわけがないと思っていたので興味はなかったのだが，押し入れを整理する必要にせまられて，これもスーパーで買ってみた。

掃除機の吸引力を使ってフトンや毛布を真空パックし，約３分の１の厚さに圧縮する。使わないあいだ小さくして収納しようというアイデアは，いかにも賢い考えのように思える。実行するにあたって，注意書きを熟読した。なんとやはり，フトンの復元については万全ではないと断わりがある。それをきっかけに考えた。

フトンの体積を減らす目的は，そのぶん別のものを収納しようということだ。たとえば冬のフトンを夏のあいだ圧縮しておくとする。あいたスペースに新しくものを買ったり，ほかのものを入れたりする。が，冬がきてフトンを元にもどそうとすると，押し入れにはそのスペースがない。フトンの厚さが増すぶん，同じ体積だけ夏のものを圧縮できればよいが，夏の寝具はもともとたいしたボリュームもない。

つまり，圧縮袋をたくさん買ってフトンを何枚もちぢめるほど，押し入れにはものをたくさん入れられるようになる。そして元にもどしたとき，押し入れに入らないフトンがふえる。こうして圧縮袋の目的はわからなくなり，押し入れの新しい荷物になった。



illustration：Kyoko Takazawa

手のひらに堅い本を

便利そうなもの，いままで見たことがなかったようなものがあらわれると，それだけで心楽しいものがある。そして，いざ使ってみて，ほんとうに便利と感じるかどうかは，その人の考えかたや習慣で，ずいぶんちがうものだと思う。

高枝切りのハサミも役にたつ人はたくさんいるだろうし，散髪用の肩かけだって，庭のある家なら，隅のほうにいって裏がえしにすればおおよそはきれいになる。フトン圧縮袋も，使うアテがほとんどない場合や，引っ越しなどで荷物の体積をへらしたりするには便利かもしれない。

通信販売もアイデア商品も，あまり高い値段のものはない。そのかわり，思いつきだけで商品化されたようなものもあるが，買う人が，慎重に自分が使う場合をイメージしてからもとめればいい。

もうちょっと高価になると，買うときに考える時間も長くなる。

いよいよ電子書籍が売り出されたそうだ。これは通信販売ではないけれど，テレビや雑誌で宣伝されてみんなが買いもとめるのだから，通信販売と似たようなものだ。

手のひらにのるサイズで，液晶画面，単行本３冊くらいまで記憶できるそうだ。片手で操作でき，印をつけておくと，すぐにそのページを呼び出せる。旅行などに何冊も本を持ち歩く必要がなくなる。

ＮＥＣの「デジタルブック」の場合，本体が29,800円，本の内容を読みとる装置が12,800円だそうだ。これに書物の内容をおさめたフロッピーが3,000円くらい。このくらいの支出になると，ちょっと慎重になる。

液晶画面で延々と読書（読字）をつづけるのはどんな気分だろう。紙の色と書体の調和の妙，本の重さや手ざわりへの郷愁を断ちきれるだろうか。どこまで読んだかという量的な実感もなくなる。ただし便利とは，何かが省かれることだ。一度もためさないで批判するべきではない。

先日，通信販売でもうひとつ，キャスターつきのハンガースタンドを買った。帰宅したあとの衣類をかけておくのが目的だった。最大量で24着かけられ，幅も高さも調節でき，どこへでも移動できる。これなら便利まちがいないと思った。

はじめのうちはコートや上着がかけられていたが，それはまもなく，ごくしぜんに洗濯ものの室内干しスタンドになった。目的とはちがう使いかただが，いまやわが家でもっとも便利な生活用品のひとつだ。

名目やスタイルでものを選んでも，けっきょく，必要と使いやすさが便利を生むようだ。便利は，まずは３日つづかなくては長生きできないだろう。

モニタの応募方法

モニタ募集

希望するモニタ記号を，とじ込みのアンケートはがきの左下のスペースまたは官製はがきに記入してお申し込みください。応募の際に使用環境について明記する必要はありませんが，当選された方には，モニタとして使用ののち，レポートを提出していただきます。締め切りは1994年2月18日の到着分までとし，当選者の発表は1994年4月号で行います。また，雑誌公正競争規約の定めにより，当選された方はこの号のほかの懸賞には当選できない場合がありますので，ご了承ください。

A

マイクロウェアシステムズ　☎03(3257)9000

OS-9ソフトウェアパッケージセット

OS-9/X68030 V2.4.5　X68030用　25,000円(税別)
Ultra C & Professional Pack V1.1　X680x0用　45,000円(税別)
Technical Tool Kit V2.4.5　X68030用　20,000円(税別)
X Windows V11.5　X68030用　30,000円(税別)

3.5+5"2HD版　1名



バージョンアップされたOS-9。関連ツールも次々と発売されていますが，現在発売中の製品をセットでモニタしていただきます。X68030，メモリ8Mバイト以上，ハードディスク約100Mバイト以上が必要です。

B

マグマソフト　☎0992(68)2286

Y300-A ver1.1

X68000用　5"2HD版　34,800円(税別)　1名



1月号のSOFTWARE INFORMATIONでもご紹介した版下作成支援プログラムです。バージョンアップで操作性が向上しました。

愛読者プレゼント

プレゼントの応募方法

とじ込みのアンケートはがきの該当項目をすべてご記入のうえ，希望するプレゼント番号をはがき右下のスペースにひとつ記入してお申し込みください。締め切りは1994年2月18日の到着分までとします。当選者の発表は1994年4月号で行います。また，雑誌公正競争規約の定めにより，当選された方はこの号のほかの懸賞には当選できない場合がありますので，ご了承ください。

1

アスキー　☎03(5351)8111

X68000ログイン ソフトウェアコンテスト 傑作ゲーム選

X68000用　5"2HD版

4,980円(税込)

3名



1月号のゲームレビューでご紹介したDISK&BOOK。LOGiN誌上で行われたコンテストの入選作品12本が収録されています。

2

スピタル産業　☎03(3251)2918

マウス　X68000用　9,800円(税別)

3名



1月号のペンギン情報コーナーでご紹介したX68000用マウスです。5段階のカウント切り替えが可能で，手ぶれを50%自動補正するファジーリバイス機能つき。

12月号Oh!X6周年記念愛読者プレゼント当選者

1 SX-WINDOWイラスト集VOL.1一般実用編　(宮城県)島田 武　(神奈川県)政所義信　**2** SX-WINDOWイラスト集VOL.2行事・四季編　(東京都)大林光明　(兵庫県)児島 匡　**3** XBAStoC CHECKER PRO-68K　(山形県)石澤敏博　(埼玉県)平田恭敏　**4** CARD PRO-68Kシステム手帳リフィル集　(東京都)松本征二　(京都府)阪長俊之　**5** CARD PRO-68K ver2.0パーソナルプログラム集　(東京都)鈴木郁哉　(大阪府)田中一匡　**6** CARD PRO-68K ver2.0ビジネスプログラム集　(滋賀県)平井智仁　(大阪府)匹野義博　**7** GRAPHIC LIBRALY VOL.2　(千葉県)永井正昭　(富山県)室谷由久　**8** GRAPHIC LIBRALY VOL.3　(埼玉県)石山靖高　鈴木条路　**9** CANVAS PRO-68Kドローグラフィックライブラリ VOL.1　(新潟県)尾崎康弘　(群馬県)佐藤慎吾　**10** CANVAS PRO-68Kドローグラフィックライブラリ VOL.2　(奈良県)塩野佳文　(兵庫県)原田大次郎　**11** MUSIC PRO-68K,MUSIC PRO-68K［MIDI］ソングライブラリ<101曲集>　(埼玉県)森山裕史　(福岡県)胤森栄二　**12** 沙羅曼蛇　(埼玉県)山下竜二　(東京都)黒沢一仁　(滋賀県)北畠 駿　(福岡県)安部一馬　**13** 熱血高校ドッジボール部　(青森県)前田桂史　(栃木県)福田安章　(東京都)関根隆仁　(愛知県)壁谷善嗣　**14** PAC・MANIA　(東京都)山上秀景　(愛知県)増井 隆　(兵庫県)中込 浩　(鹿児島県)前田多門　**15** SUPER HANG-ON　(福島県)吉仲直子　(静岡県)杉山和弥　望月利修　(三重県)長谷川常吉　**16** Thunder Blade　(北海道)鈴木勇　(福島県)戸辺 靖　佐井川泰治　(京都府)吉山伸司　**17** V'BALL　(埼玉県)杉山洋之　(東京都)間島恒己　(奈良県)小笹由貴　(大分県)中村 忍　**18** ダウンタウン熱血物語　(宮城県)高橋光一　(群馬県)黒澤典義　(広島県)鈴木篤志　(徳島県)松島俊彦　**19** 中華大仙　(北海道)益山直人　(埼玉県)瀬尾達人　(神奈川県)安西由征　(兵庫県)臼田幸生　**20** ダッシュ野郎　(千葉県)神田貴則　(愛知県)武田英明　(兵庫県)小池哲也　(大分県)河野真司　**21** BONANZA BROS.ボナンザブラザーズ　(秋田県)内藤祐希　(千葉県)金子聡史　(東京都)中村直哉　(大阪府)有馬健

(敬称略)

以上の方々が当選しました。商品は順次発送いたしますが，入荷状況などにより遅れる場合もあります。

こちらシステムX探偵事務所

FILE-IX

旋律を作る

Tamura Kento 田村 健人

今月は，なんとシステムX探偵事務所の社歌を作ります。といっても曲を作るのはX68000。プログラムを担当するのは，探偵事務所初登場の田村氏です。さて，無事に曲は完成するのでしょうか。

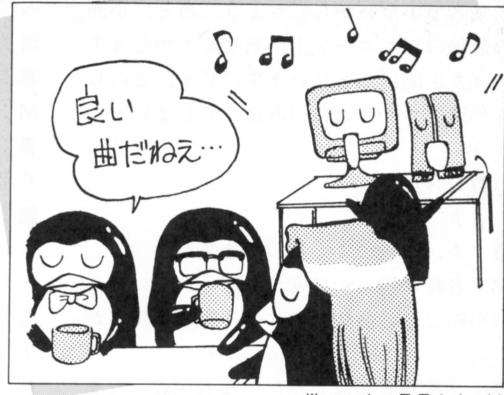


illustration:T.Takahashi

琴張春香(以下春)：だいたい学校には校歌ってあるじゃない？
マスター(以下M)：ええ，そうですね。私のかよっていた小学校の校歌は途中で女子パートと男子パートがあるんですよ。調子にのって歌っていると，思わず女子のパートまで歌ってしまってひんしゅくものでした。
琴張護(以下護)：私は歌うことがあまり好きではありませんので，どうも校歌は嫌いでしたね。校歌の存在意義もよくわかりませんし。
春：ないよりはあったほうが楽しいんじゃない？
M：それで，どうしたんですか？　いきなり校歌の話なんて。
春：社歌っていうのもあるわよね。
M：そりゃまあ，あるところにはあるでしょうねぇ。……ま・さ・か。
春：うふふ，ほしいと思わない？
護：いえ，社歌なんてなければないでよろしいのでは……。
春：護ちゃん！
護：社歌，欲しいですね。
M：護さん，あんた弱いねえ。
春：3人中，まず2人は賛成。過半数ってことで，「システムX探偵事務所のテーマ」作成決定ね。
M：過半数ったって，私が責任者……。
春：うるさいわねっ！
M：いっておきますが，プロの作曲家に頼む予算なんてありませんよ。
春：3人とも作曲なんかできないし……。どうしようかしら。護ちゃん，なんとかして。
護：なんとかしろといわれましても……。あ，知り合いに音楽関係のサークルに入っていた人がいるんで，彼に頼んでみましょう。
（数時間後）
♪ピンポーン
田村健人(以下健)：こんばんわー。琴張氏に呼ばれて来ました～。
護：あ，健人くん来てくれてありがとう。実はですね……ということで。
健：えっ，私も作曲はできませんよ。
M：まー，私たちよりは音楽の知識はあるでしょ。はっ，ここに1台のX68000が！
健：なーるほど。コンピュータに作曲をさせてしまおうっていうことですね。

自動作曲

健：まず，コンピュータによる自動作曲というと，音程と音長を乱数で決める方法を思いつきますね。
M：そうそう。「乱数なんだから，何回か実行すればいつかは素晴らしい曲が演奏されるはずだ」とか思いながら実行しても，ちっとも曲にならないんですよね。
健：この方法では音楽における重要なファクターを完全に無視しているんです。
護：その重要なファクターというのはなんですか？
春：コード進行ね。
健：そう，コード進行です。実際に人間が作曲をする場合でも，コード進行を決めてから旋律を作るのが基本なんだそうです。
護：「だそうです」というのは頼りないですね。
健：そんなこといったって，私だって作曲なんてしたことないんですから。
春：大丈夫なの～？
健：なんとかなるんじゃないですか？　で，実はもうひとつ「リズム」についても考えなくちゃいけないんですね。
M：リズム？　ドラムとかのことですか？
健：いえ，そこまで細かい話ではなくて，最低レベルの「小節で区切られる」ということです。乱数で音長を決めた場合，小節の枠にまったく無関係になってしまいます。
春：たいていの曲は2～4小節ぐらいで区切りがつくわね。
健：コード進行と旋律の関係ですが，コードに含まれる音を適当に並べたものが旋律です。
M：そんなに単純なものなんですか？
健：嘘です。
M：うっ。
健：もちろんコード上ではない音も使います。えーと，とりあえずここまでのところでプログラムにして，結果を聞いてみましょう。コード進行を入力して，コード上の音が多くなるような旋律を出力します。リズムについてはあと回しにして，いまのところ8分音符に固定します。
（数十分後）
健：ふう，こんなもんかな。実行してみてください。
護：コード進行を入力するんですよね？　でもコード進行なんてわかりませんよ。
健：とりあえず C→F→G→C の4小節でやってみてください。
護：わかりました。では……
♪♪
M：へえ～，音長と音程を乱数で決める方法に比べると，ものすごくまともになっています。
護：8分音符固定でも聞けるものですね。
春：でも，音程が上下に飛びすぎるわ。
健：しょせん，音程は乱数ですからね。これは前の音程との差を乱数で決めるようにしましょう。
春：あと，これってどうも小節の区切りがぼやけているような気がする。
健：いまのうちにことわっておきますが，私はこれ以後自分の経験に基づいて説明します。だから音楽理論から外れるかもしれませんので，必ず疑いながら聞いてください。固定音長なのに小節の区切りがはっき

りしないのは，各小節の頭の音がコード上に乗っていないからでしょう。あと，小節の最後の音もコード上に乗せるとかなりすっきりと聞けると思います。じゃ，これらも早速プログラムに組み込んでしまいましょう。

（数分後）

♪♪

護：あ，本当によくなってますね。

M：音程が滑らかに変化し，それでいてこの小節ごとの歯切れのよさ。実に気持ちがいい。

春：「美味しんぼ」みたいな表現ね。たまに音程が上か下にいきっぱなしになるわ。

健：前の音程との差が負か正に偏ってしまうときですね。とりあえず，いちばん初めの音から上下1オクターブ以内に収まるようにしてしまいましょう。あと，指定したコードも鳴らすようにしましょうね。

春：コードを入れるんなら，ベースも入れたら？

健：そうですね。ベースとしてコードの根音を4分音符で鳴らしておきましょう。

（数分後）

♪♪

春：コードとベースを加えるだけでかなり音楽性が増すのね。

護：旋律の音程がまだ少しおかしいと思うのですが。

健：そうですね。なんでだと思います？

M：音程を決めるのにコード上の音かどうか以外の要素もあるんですか？

健：ありますねえ。しかも，普通みなさんはコードよりもそっちの概念のほうが身近だと思いますよ。「ファミリーレストラン」って略してなんていいます？

護：唐突な質問ですね。

春：わたしは「ファミレス」。

M：「ファミレ」っていいますね。

春：あ，そうか。「ドレミファソラシド」。音階のことね！

健：そそそ，そーです。おそらく音階のほうがコードよりも優先されるべきなんでしょうね。普通の曲でも，音階，つまり調から外れる音は，コードから外れる音よりは少ないと思います。

M：じゃあどうするんですか？　コード上の音なのに音階に乗っていない音ってざらにあるでしょ？

春：ハ長調でC7だとB♭が音階に乗らないわ。

健：このあたり，どうするのがベストなのかよくわからないんですよ。とりあえず，コードから外れる音は必ず音階に乗るようにします。逆にいえば，音階から外れていても，その音はコード上の音であるということになります。ところで，リズムというか音長のことですが，なにかうまい方法はありませんか？

護：いまのままの8分音符固定でもいいじゃないですか。十分聴けますよ。

健：そういうわけにもいかないでしょう。

図1　乱数による汚いリズム

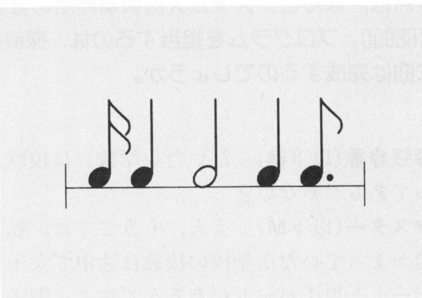
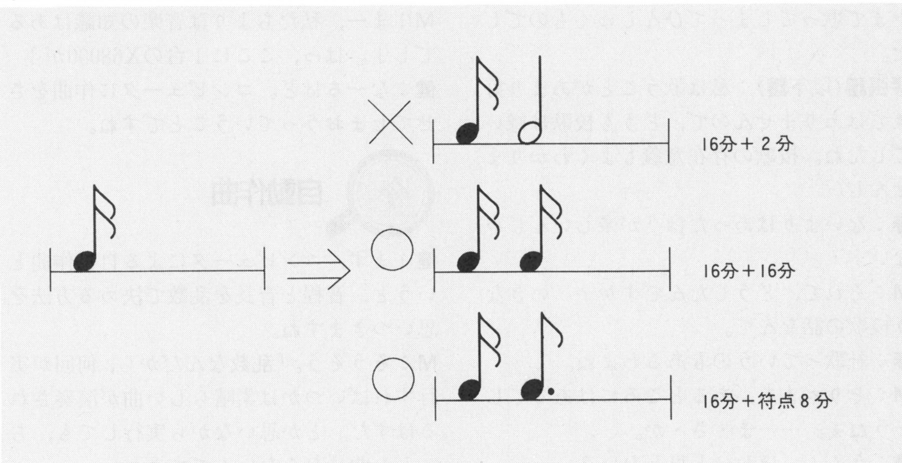


図2　小節の頭に16分音符がある場合



プログラムの使い方

コンパイルにはGCCを実行できる環境が必要です。リストを打ち込んで，たとえば，

A＞gcc -O zmkph.c -lfloatfnc

とコンパイルしてください。

実行するにはX680x0とfloat?.xとZ-MUSICシステム（とくにバージョンは問いません）が必要です。

実行するときには設定ファイル（拡張子：zph)が必要になります。エディタなどでサンプルのようなコード進行を書き込んでください。

A＞zmkph sample.zph

とするとsample.zmsを出力します。結果をすぐに聞きたいときは，上記の代わりに，

A＞zmkph sample.zph || zp sample.zms

または，Z-MUSICが常駐した状態で，

A＞zmkph sample.zph || copy sample.zms opm

とすればいいでしょう。

設定ファイルは，基本的にコード進行を羅列したテキストファイルです。コードは，メジャー・マイナー・メジャー7・マイナー7のみです。根音は大文字で書いてください。

C C+ D- D D+ E- E F F+ G- G G+ A- A A+ B- B

+-は#b(Bの小文字)で代用できます

マイナーコードでは根音に続けてm(小文字)を書きます。メジャー7は根音のあとに7(半角)を書きます。もうおわかりでしょうが，マイナー7はm7です。コードをタブ・スペースで区切って羅列すると，それぞれが1小節に対応します。

タブ・スペースで区切らない場合は，

C7F	C7,F：それぞれ2分音符
C7FG	C7：2分音符　F,G：4分音符
C7FGC	C7,F,G,C：4分音符

となります。

ピリオドで始まる行でコード進行以外の値を設定できます。

.scale Cm	：音階をハ短調にする
.ulimit A5	：オクターブ5のA以下の音を使う
.llimit Bb3	：オクターブ3のBｂ以上の音を使う

#で始まる行，空行は無視されます。

なお，このプログラムで作った曲の著作権は利用者が保持するものとなります。

リスト2　SAMPLE.ZPH

```
#  設定ファイルのサンプル
# で始まる行はコメントです
# 音階を指定します
# デフォルトは C です
.scale D

D D
D D
G G
G G
A A
D D
```

だいいち，私のプライドが許しません。ちなみにプライドっていうのはコカコーラボトラーズの清涼飲料水とはなんの関係もありません。

M：乱数だと問題があるのはなぜでしたっけ？

健：音長が小節の枠に無関係になるって理由です。

M：じゃあ適当に，16分音符の何倍かを乱数で決めて，小節をまたぎそうになったら無理やり切り詰めてしまえばいいのではないですか？

春：それで美しい旋律になるとは思えないわ。乱数だと図１のようなリズムも平気で出力されるわね。こんなのいや。

M：それもそうですね。

春：ちょっと考えたんだけど，小節の頭に16分音符がきた場合，その次に２分音符がくることはまずないわよね？

健：そうですね。この場合，次に来るのが16分音符か符点８分音符だとしっくりするかな。(図２)

春：こう考えると，ちょっと規則性が見えてこない？

健：やってみましょう。

（数分後）

健：うるうる。

護：どうしたんですか？

健：音長の規則性はわかっているんですが，時間の流れに添って処理していくいままでのプログラムでは規則を簡潔に書くのがものすごく難しいんですよ。

M：どう対処しました？

健：16分音符を単位としたので，１小節のうちの埋まっている割合で16通りに分岐してしまいました。うう。あまりにも美しくないプログラムを書いてしまって泣けてきます。しくしく。

春：まあまあ，気を取り直して実行してみましょ。

♪♪

M：おお，すごい！　曲になってる。

健：ふっ，まかせなさあい！

春：あ，元気になった。

健：これぐらいのクオリティならまあ実用に堪えるでしょう。

護：でも何回実行してもあまり雰囲気が変わらないような気がします。

健：コード進行が同じなら曲の雰囲気も同じようなものになるってことで納得してください。

システムX探偵事務所のテーマ

M：Cで作ってたんですね。

護：この程度の処理ならBASICで書けませんか。

健：X-BASICは嫌いなんです。

護：なぜです？

健：バックスクロールができないじゃないですか。

M：バックスクロールできるBASICなんてあるんですか？

健：N88-BASICはできますよ。あと，編集操作がEmacs系に慣れきってしまっていて……。

M：Emacsを使いたかったら，Emacsでプログラムを作ってX-BASICで実行すればいいじゃないですか。

健：そんなの面倒ですよ。

春：怠慢！　怠慢！

健：「タイマン」っていうと，結婚式をやるにはちょうどいいっていう日ですか？

M：それをいうなら「大安」でしょ。ま，とにかく「システムX探偵事務所のテーマ」を作りましょうか。誰かコード進行を考えてください。

一同：……。

健：あ，誰もコード進行を作れない。これから本誌で連載していた「Creative Computer Music入門」を読み返して勉強しますか。

春：え～，結局作曲の勉強をしなきゃいけないの～。　　　　(つづく)

リスト１

```
/*          Oh!X 1994.2 システムX探偵事務所
            旋律生成プログラム
                                          by けんと */

/* NOTICE
            このプログラムを用いて出力した曲は、
       設定ファイルを書いた利用者の著作物とし
       ます。
*/

#include          <stdlib.h>
#include          <stdio.h>
#include          <string.h>
#include          <stdarg.h>
#include          <time.h>

#define CC        0
#define CCS       1
#define CD        2
#define CDS       3
#define CE        4
#define CF        5
#define CFS       6
#define CG        7
#define CGS       8
#define CA        9
#define CAS       10
#define CB        11

#define L16       (192/16)
#define L8        (L16*2)
#define L8P       (L8*3/2)
#define L4        (L8*2)
#define L4P       (L4*3/2)
#define L2        (L4*2)
#define L2P       (L2*3/2)
#define L1        192
enum { FALSE, TRUE };
                                    /* 0～99 を返す */
#define R100()  (rand()*100/(RAND_MAX+1))

typedef struct {
     short m;                       /* 0:C ～ 11:B */
     char min;                      /* 1: minor */
     char sev;                      /* 1: 7th */
} code;

typedef struct {
     code c[4];
} 小節;


int PTOPONCODE = 100;  /*フレイズの最初の音がコード上である確率*/
int PONCODE = 70;      /*音がコード上である確率*/
int P16OR8P = 80;      /*符点8分音符が余っている時に16分音符にする確率*/
int P16OR8 = 80;       /*8分音符が余っている時に16分音符にする確率*/

int 音階 = CC;                       /* 音階を表す */
int 短調 = FALSE;                    /* 0:長調 1:短調 */
int pren = (4+1)*12+CA;               /* 前に出した音程 初期値A4 */
int 音域上 = (5+1)*12+CA;
int 音域下 = (3+1)*12+CA;
FILE* pf;
int bVerbose = FALSE;
int InstrumentType = 0;
小節 sc[128];

const char* s[] = {
     "c",  "c+", "d",  "d+", "e",  "f",
     "f+", "g",  "g+", "a",  "a+", "b",
};
const short char2note[7] = {
     9,11,0,2,4,5,7,
};


int mkph( 小節* p, int n );
int mkcode( 小節* p );
int getnote( code* pc, int oncode, int prevnote );
int oncodep( code* pc, int note );
int onscalep( int note );
int inzonep( int note );
int outnote( int note, int len );



void vmes( char* pmes, ... ) {
     if ( bVerbose ) {
          va_list pparam;
          va_start( pparam, pmes );
          vfprintf( stderr, pmes, pparam );
          va_end( pparam );
     }
     return;
}



int main( int argc, char** argv ) {
     unsigned short seed;
     int i, argfn = 0;
     FILE* pfc;
     char sFname[100];
     char* p;
```

```
	char sRead[1024];

	seed = time( NULL );

	if ( argc == 1 ) {
	    fprintf( stderr,
		    "Usage: zmkph [options] inputfile.zph¥n"
		    "options:¥n"
		    "	    -v	    verbose¥n"
		    "	    -s num  seed of random¥n"
		    "	    -t type type==(cm6|opm)¥n" );
	    return 2;
	}
					/* オプション解析 */
	for ( i=1; i<argc; i++ ) {
	    if ( argv[i][0] == '-' ) {
		int j = 1;
		while ( argv[i][j] && j ) {
		    char* pc;
		    switch ( argv[i][j] ) {
		      case 's':
			if ( argv[i][j+1] ) {
			    pc = &(argv[i][j+1]);
			} else {
			    pc = argv[i+1];
			    i ++;
			    if ( i == argc ) {
				fprintf( stderr, "¥n-s の引き数があ
りません¥n" );
				return 2;
			    }
			}
			j = -1;
			seed = atoi( pc );
			break;
		      case 't':
			if ( argv[i][j+1] ) {
			    pc = &(argv[i][j+1]);
			} else {
			    pc = argv[i+1];
			    i ++;
			    if ( i == argc ) {
				fprintf( stderr, "¥n-t の引き数があ
りません¥n" );
				return 2;
			    }
			}
			j = -1;
			if ( !strncmp( pc, "cm6", 3 ) )
					InstrumentType = 1;
			if ( !strncmp( pc, "CM6", 3 ) )
					InstrumentType = 1;
			if ( !strncmp( pc, "opm", 3 ) )
					InstrumentType = 0;
			if ( !strncmp( pc, "OPM", 3 ) )
					InstrumentType = 0;
			break;
		      case 'v':
			bVerbose = TRUE;
			break;
		      default:
			fprintf(stderr,"¥n不正な option です¥n");
			return 2;
		    }
		    j ++;
		}
	    } else {			/* ファイル名のようだ */
		argfn = i;
	    }
	}

					/* 乱数シードの設定 */
	srand( seed );
	vmes( "疑似乱数シード: %hu¥n", seed );

					/* 読み込みファイルオープン */
	strcpy( sFname, argfn ? argv[argfn] : "foo.zph" );
	vmes( "Open %s ...¥n", sFname );
	if ( (pfc = fopen( sFname, "rt" )) == NULL ) {
	    fprintf( stderr, "ファイルをオープンできません¥n" );
	    return 1;
	}

					/* 出力ファイルオープン */
					/* 拡張子を .zms に変えるだけ */
	p = strchr( sFname+strlen( sFname )-4, '.' );
	if ( !p )  p = sFname+strlen( sFname );
	strcpy( p, ".zms" );
	vmes( "Open %s ...¥n", sFname );
	if ( (pf = fopen( sFname, "wt" )) == NULL ) {
	    fprintf( stderr, "ファイルをオープンできません¥n" );
	    return 1;
	}

	fprintf( pf, ".comment 作曲:X680x0¥n" );
	fprintf( pf, "/ 乱数シード %hu¥n", seed );
	fprintf( pf, "(i)¥n" );
	if ( InstrumentType == 1 ) {
	    fprintf( pf, "(m01,2000)(aMIDI11,1)¥n" );
	    fprintf( pf, "(m02,2000)(aMIDI12,2)¥n" );
	    fprintf( pf, "(m03,2000)(aMIDI13,3)¥n" );
	    fprintf( pf, "(m04,2000)(aMIDI14,4)¥n" );
	    fprintf( pf, "(m05,2000)(aMIDI15,5)¥n" );
	    fprintf( pf, "(m08,2000)(aMIDI16,8)¥n" );
	    fprintf( pf, "(t1) @1  @v120 @u100 o%1d¥n", pren/12
-1 );
	    fprintf( pf, "(t2) @37 @v100 @u70 o4¥n" );
	    fprintf( pf, "(t3) @37 @v100 @u70 o4¥n" );
	    fprintf( pf, "(t4) @37 @v100 @u70 o4¥n" );
	    fprintf( pf, "(t5) @37 @v100 @u70 o4¥n" );
	    fprintf( pf, "(t8) @29 @v115 @u100 o4¥n" );
	} else {
	    fprintf( pf,
					/* 68snd.zms より1番 */
	    "(v1,0¥n"
	    "/   AF  OM  WF  SY  SP PMD AMD PMS AMS PAN¥n"
	    "    58, 15,  2,  0,220,  0,  0,  0,  0,  3,  0¥n"
	    "/   AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AME¥n"
	    "    28,  4,  0,  5,  1, 37,  2,  1,  7,  0,  0¥n"
	    "    22,  9,  1,  2,  1, 47,  2, 12,  0,  0,  0¥n"
	    "    29,  4,  3,  6,  1, 37,  1,  3,  3,  0,  0¥n"
	    "    15,  7,  0,  5, 10,  0,  2,  1,  0,  0,  1)¥n"
					/* 68snd.zms より9番 */
	    "(v29,0¥n"
	    "/   AF  OM  WF  SY  SP PMD AMD PMS AMS PAN¥n"
	    "    58, 15,  2,  0,150,  0,  0,  0,  0,  3,  0¥n"
	    "/   AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AME¥n"
	    "    31, 13,  1,  4, 15, 32,  1,  0,  7,  0,  0¥n"
	    "    31, 11,  1, 10, 15, 55,  1,  4,  5,  0,  0¥n"
	    "    31, 11,  1, 10, 15, 29,  0,  0,  2,  0,  0¥n"
	    "    31, 11,  1,  8, 15,  0,  1,  0,  3,  0,  1)¥n"
					/* 68snd.zms より21番 */
	    "(v37,0¥n"
	    "/   AF  OM  WF  SY  SP PMD AMD PMS AMS PAN¥n"
	    "    58, 15,  2,  0,205, 80,  0,  2,  0,  3,  0¥n"
	    "/   AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AME¥n"
	    "    30,  1,  0,  1,  1, 30,  3,  0,  2,  0,  0¥n"
	    "    31,  1,  0,  2,  1, 38,  3,  2,  3,  0,  0¥n"
	    "    30,  1,  0,  1,  1, 48,  1,  1,  3,  0,  0¥n"
	    "     8,  2,  0,  6,  0,  0,  0,  1,  4,  0,  1)¥n"
);
	    fprintf( pf, "(m01,2000)(aFM1,1)¥n" );
	    fprintf( pf, "(m02,2000)(aFM2,2)¥n" );
	    fprintf( pf, "(m03,2000)(aFM3,3)¥n" );
	    fprintf( pf, "(m04,2000)(aFM4,4)¥n" );
	    fprintf( pf, "(m05,2000)(aFM5,5)¥n" );
	    fprintf( pf, "(m08,2000)(aFM8,8)¥n" );
	    fprintf( pf, "(t1) @1 @v120 o%1d¥n", pren/12-1 );
	    fprintf( pf, "(t2) @37 @v100o4¥n" );
	    fprintf( pf, "(t3) @37 @v100o4¥n" );
	    fprintf( pf, "(t4) @37 @v100o4¥n" );
	    fprintf( pf, "(t5) @37 @v100o4¥n" );
	    fprintf( pf, "(t8) @29 @v115o4¥n" );
	}

#define SKIPSP( A )	while((*A==' ')||(*A=='¥t'))A++

					/* 設定ファイルを1行ずつ読む */
	while ( fgets( sRead, 1024, pfc ) ) {
					/* 空行は無視 */
	    if ( sRead[0] == '¥n' )  continue;
					/* #で始まる行は無視 */
	    if ( sRead[0] == '#' )  continue;
					/* 各値の設定 */
	    if ( sRead[0] == '.' ) {
		char* pc = sRead+1;
		int n, t;

#define SCANNOTE(N,P)	N = char2note[*P++-'A'];	¥
			if ( (*P=='#')||(*P=='+') ) {	¥
			    N ++;  P ++;		¥
			} else if ((*P=='-')||(*P=='b')){¥
			    N --;  P ++;		¥
			}
		if ( !strncmp( pc, "scale", 5 ) ) {
		    pc += 5;  SKIPSP( pc );
		    SCANNOTE( n, pc )
		    音階 = n;
		    短調 = (*pc=='m');
		    vmes( ".scale¥n" );
		} else if ( !strncmp( pc,  "ulimit", 6 ) ) {
		    pc += 6;  SKIPSP( pc );
		    SCANNOTE( n, pc );
		    sscanf( pc, "%d", &t );
		    音域上 = (t+1)*12+n;
		    vmes( ".ulimit¥n" );
		} else if ( !strncmp( pc,  "llimit", 6 ) ) {
		    pc += 6;  SKIPSP( pc );
		    SCANNOTE( n, pc );
		    sscanf( pc, "%d", &t );
		    音域下 = (t+1)*12+n;
		    vmes( ".llimit¥n" );
		} else if ( !strncmp( pc,  "poncode", 7 ) ) {
		    pc += 7;  SKIPSP( pc );
		    sscanf( pc, "%u", &PONCODE );
		    vmes( ".poncode¥n" );
		} else if ( !strncmp( pc,  "ptoponcode", 10 )){
		    pc += 10;  SKIPSP( pc );
		    sscanf( pc, "%u", &PTOPONCODE );
		    vmes( ".ptoponcode¥n" );
		} else if ( !strncmp( pc,  "lentable", 8 ) ) {
		    pc += 8;  SKIPSP( pc );
		    /* 未実装 */
		} else {
		    fprintf(stderr,"Warning: 未定義の設定命令です¥n")
;
		}
	    } else {			/* コード進行 */
		char* pc = sRead;
		int n = 0;
		int i;
		int c;
		int min;
		int sev;

		SKIPSP( pc );
		while ( *pc != '¥n' ) {
		    if ( *pc>='A' && *pc<='G' ) {

#define GETCODE()	SCANNOTE(c,pc)			¥
			min = 0;			¥
			sev = 0;			¥
			if ( *pc == 'm' ) {		¥
			    min = 1;  pc ++;		¥
			}				¥
			if ( *pc == '7' ) {		¥
			    sev = 1;  pc ++;		¥
			}

			GETCODE()
			sc[n].c[0].m = c;
			sc[n].c[0].min = min;
```

▶先日，家のドアに名刺がはさんであった。見ると「中国男」と書いてあったので驚いたが，よく見ると「広中　国男」さんでした。　種井　裕光(17)愛知県

```
                    sc[n].c[0].sev = sev;
                    sc[n].c[1] = sc[n].c[0];
                    sc[n].c[2] = sc[n].c[0];
                    sc[n].c[3] = sc[n].c[0];
                    if ( *pc>='A' && *pc<='G' ) {
                        GETCODE()
                        sc[n].c[2].m = c;
                        sc[n].c[2].min = min;
                        sc[n].c[2].sev = sev;
                        sc[n].c[3] = sc[n].c[2];
                        if ( *pc>='A' && *pc<='G' ) {
                            GETCODE()
                            sc[n].c[3].m = c;
                            sc[n].c[3].min = min;
                            sc[n].c[3].sev = sev;
                            if ( *pc>='A' && *pc<='G' ) {
                                GETCODE()
                                sc[n].c[1] = sc[n].c[2];
                                sc[n].c[2] = sc[n].c[3];
                                sc[n].c[3].m = c;
                                sc[n].c[3].min = min;
                                sc[n].c[3].sev = sev;
                            }
                        }
                    }
                }
                SKIPSP( pc );
                n ++;
            }                       /* end of while */
            mkph( sc, n );
            for ( i=0; i<n; i++ )
                mkcode( &(sc[i]) );
        }
    }                               /* end of while */

    fprintf( pf, "¥n(p)¥n" );
    fclose( pf );
    fclose( pfc );

    return 0;
}



int mkcode( 小節* p ) {
    int n,i;
    for ( i=0; i<4; i++ ) {
        n = (4+1)*12+p->c[i].m;
        fprintf( pf, "(t2) %s4&¥n", s[p->c[i].m] );
        fprintf( pf, "(t8) %s4¥n", s[p->c[i].m] );
        n += 4-(p->c[i].min);
        fprintf( pf, "(t3) o%1u%s4&¥n", n/12-1, s[n%12] );
        n = (4+1)*12+p->c[i].m +7;
        fprintf( pf, "(t4) o%1u%s4&¥n", n/12-1, s[n%12] );
        if ( p->c[i].sev ) {
            n += 3;
            fprintf(pf, "(t5) o%1u%s4&¥n", n/12-1,s[n%12]);
        } else {
            fprintf( pf, "(t5) r4¥n" );
        }
    }
    return 0;
}



int mkph( 小節* p, int n ) {
    int totalstep = 0;
    int note;
    int step;
    code* pc;
    static const int 音長表[20] = {
        L16, L8,  L8,  L4,  L4,  L4,  L4P, L2,  L2P, L1,
        L8,  L8,  L8,  L4P, L8,  L8,  L8P, L8,  L2,  L8,
    };

    fprintf( pf, "(t1) " );
    while ( totalstep < 192*n ) {
        int oncode;
                                /* 音程を決める */
        pc = &p[totalstep/192].c[totalstep%192/48];
        oncode=R100()<(!totalstep ? PTOPONCODE : PONCODE);
        note = getnote( pc, oncode, pren );
                                /* 音長を決める */
        switch ( ((192*n-totalstep)%192)/L16 ) {
                                /* ここらへんは同じ処理が多数あるので */
                                /* もっと短く書くことが可能です */
                                /* 拡張のし易さを考えて敢えて冗長に */
                                /* 書いています */
          case 0:                       /* 全音符 */
            step = 音長表[R100()/5];
            break;
          case 15:                      /* 2分+4分+8分+16分 */
            step = R100()<P16OR8P ? L16 : L8+L16;
            break;
          case 14:                      /* 2分+4分+8分 */
            step = 音長表[R100()/5];
            break;
          case 13:                      /* 2分+4分+16分 */
            step = L16;
            break;
          case 12:                      /* 2分+4分 */
            step = 音長表[R100()/5];
            break;
          case 11:                      /* 2分+8分+16分 */
            step = R100()<P16OR8P ? L16 : L8+L16;
            break;
          case 10:                      /* 2分+8分 */
            step = 音長表[R100()/5];
            break;
          case 9:                       /* 2分+16分 */
            step = L16;
            break;
          case 8:                       /* 2分音符 */
            step = 音長表[R100()/5];
            break;
          case 7:                       /* 4分+8分+16分 */
            step = R100()<P16OR8P ? L16 : L8+L16;
            break;
          case 6:                       /* 4分+8分 */
            step = 音長表[R100()/5];
            break;
          case 5:                       /* 4分+16分 */
            step = L16;
            break;
          case 4:                       /* 4分音符 */
            step = 音長表[R100()/5];
            break;
          case 3:                       /* 8分+16分 */
            step = R100()<P16OR8P ? L16 : L8+L16;
            break;
          case 2:                       /* 8分音符 */
            step = R100()<P16OR8 ? L16 : L8;
            break;
          case 1:                       /* 16分音符 */
            step = L16;
            break;
          default:                      /* 有り得ないはずだが */
            fprintf( stderr, "¥n内部エラーです¥n" );
            return 1;
        }
                                /* コード上でない音はあまり長くしない */
        if ( !oncode && step>L4)  step = L4;

                                /* 小節の始めで必ず音が出るように */
        totalstep += step;
        if ( totalstep/192 != (totalstep-step)/192 ) {
            step -= totalstep-(totalstep/192)*192;
            totalstep = (totalstep/192)*192;
        }
                                /*  フレイズの最後の音はコード上 */
        if ( totalstep >= 192*n ) {
            note = getnote( pc, TRUE, pren );
        }

        outnote( note, step );
        pren = note;
    }
    fprintf( pf, "¥n" );
    return 0;
}


                                /* 音を取ってくる */
int getnote( code* pc, int oncode, int prevnote ) {
    int n;
    do {
        n=prevnote+(oncode ? (rand()%15)-7 : (rand()%5)-2);
    } while ( (n&~0x7f) || !inzonep( n )
            || (oncode&&!oncodep( pc, n ))
            || (!oncode&&(oncodep(pc,n) || !onscalep(n))));
    return n;
}



                                /* 音がコード上かどうかを調べる */
int oncodep( code* pc, int note ) {
    int i, n;
    n = note % 12;
    i = pc->m;
    if ( i == n )  return TRUE;
    i += 4-(pc->min);
    if ( i ==  n )  return TRUE;
    i = pc->m+7;
    if ( i == n )  return TRUE;
    if ( pc->sev ) {
        i += 3;
        if ( i == n )  return TRUE;
    }
    return FALSE;
}



                                /* 音が音階に含まれるかどうかを調べる */
int onscalep( int note ) {
    static const unsigned char scale[2][12] = {
        { 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, },
        { 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, }
    };
    int n;
    n = note % 12;
    n -= 音階;
    if ( n < 0 )  n += 12;
    return scale[短調][n];
}



                                /* 音域に入っているか */
int inzonep( int note ) {
    if ( note < 音域下 )  return FALSE;
    if ( note > 音域上 )  return FALSE;
    return TRUE;
}


int outnote( int note, int step ) {
    int preo = pren/12-1;
    int curo = note/12-1;
    if ( preo != curo ) {
        if ( preo < curo )
            fprintf( pf, "<" );
        else
            fprintf( pf, ">" );
    }
    fprintf( pf, "%s", s[note%12] );
    if ( 192%step ) {
        fprintf( pf, "*%1u", step );
    } else {
        fprintf( pf, "%1u", 192/step );
    }
    return 0;
}
```

▶X68000の新機種は，CD-ROM標準装備がいいなあ。　　井口　玄(19)大阪府

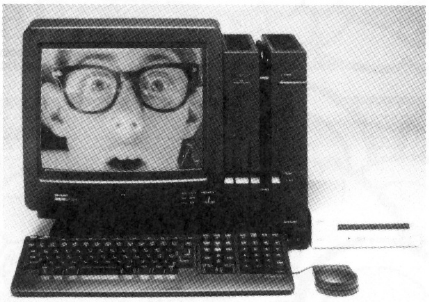

ANOTHER CGWORLD
COPY ! COPY ! COPY !
KYOKO 12 / 17 / 93
1
さーてと CGでも
つくるかね〜
2
あれ!
どーしたの?
3
ひとりで遊んで
いても、ちっとも
楽しくない
んだって…
4


そのままじっとして‥動かないで
5
COPY
COPY
COPY
COPY
6
7
8
9
みんな 同じでつまんないって？‥‥トホホ
わがままな子だねェ
10

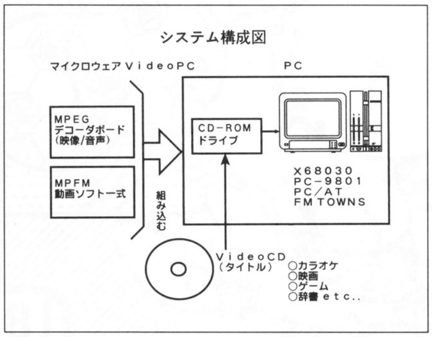
PENGUIN INFORMATION CORNER
ペ・ン・ギ・ン・情・報・コ・ー・ナ・ー

NEW PRODUCTS

OS-9/X680x0対応
Video PC for X680x0
マイクロウェアシステムズ



Video PC for X680x0



マイクロウェアシステムズは，OS-9/X680x0対応「Video PC」を発売する。

本製品は，MPEG規格で圧縮されたデータ（映像/音声）を再生するためのパッケージである。これをX680x0に組み込むことで，VideoCD（映画，カラオケ）ソフト，ハードディスクやCD-ROM内のMPEGデータファイルが扱えるようになる。パッケージには，MPEGデコーディングボード，デコーディングボード制御ソフト，CD-ROM制御ソフト，マニュアルなどが同梱されている。制御ソフトのひとつにオンスクリーンボタン機能がついている。これで「プレイ」「ポーズ」「スロー」「ステップ」「フリーズ」「スキャン」「ストップ」の7機能を画面上で操作できる。テレビ出力端子は，RGB（アナログ），NTSC端子の2種類がある。本製品を利用するためには，OSにOS-9/X680x0と倍速CD-ROMドライブが別途必要になる。また，OSにOS-9000を乗せたPC-9801，PC/AT，FM TOWNS（どれもCPU386以上）に対応したパッケージも発売する。

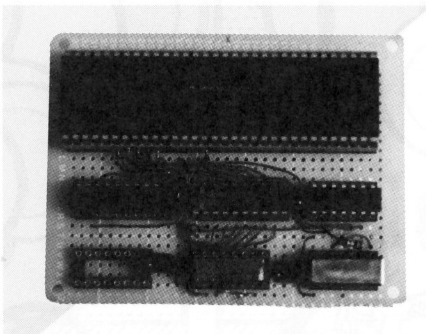
価格は，58,000円（税別）。

〈問い合わせ先〉

マイクロウェアシステムズ㈱

☎03(3257)9000

MPUアクセラレータ
H.A.R.P for MC68000
ジャスト



H.A.R.P for MC68000

ジャストはX68000用MPUアクセラレータ「H.A.R.P」を発売した。

本製品は，X68000XVI，X68000Compact XVI以外のX68000に対応したダブルクロックドMPUアクセラレータである。これは，MC68000に供給されるクロック入力をモジュール内のクロックダブラーにより2倍してコアのプロセッサに供給し，2倍のクロックスピードで動作する。外部デバイスとの電気特性がMC68000と互換性をもつので，既存のファームウェアを変更する必要がなく演算能力を向上させることが可能になった。またメモリサイクルは，3クロックと4クロックの2つのモードがある。同社のER10Sメモリボードを併用することで3クロックモードとなりアクセススピードも向上する。

価格は，29,800円（税別）。

〈問い合わせ先〉

㈱ジャスト　☎03(3706)9766

拡張I/Oボックス
ESX68L4
ジャスト

ジャストはX680x0に対応する拡張I/Oボックス「ESX68L4」を発売した。

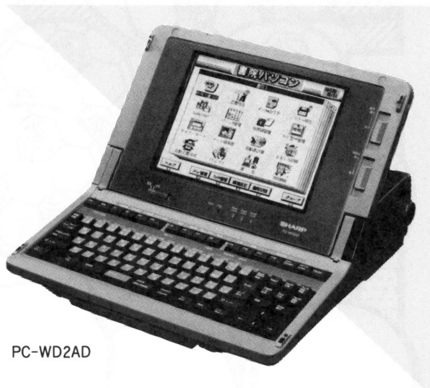
本機はX680x0に対応する増設用拡張スロットである。同梱されるものは，拡張I/Oボックス，X680x0本体とボックスを接続するためのインタフェイスカード2枚，それを接続するケーブルなど。ボックスには4個のI/Oスロットが設けられ，そのうち1個のスロットを本体との接続用に使用する。ボックス内に専用の電源（+5V/3A，+−12V/0.3A）を内蔵し，負荷の大きなボードにも対応した。

価格は，39,800円（税別）。

〈問い合わせ先〉

㈱ジャスト　☎03(3706)9766

書院パソコン
PC-WD2A/PC-WD2AD
シャープ



PC-WD2AD

シャープは書院パソコン2機種「PC-WD2A」「PC-WD2AD」を発売した。

本機は，ワープロ「書院」の機能とパソコン機能を融合させたものである。

「PC-WD2A」では，起動時にアイコンによるメニューシステムを使用している。32ビットCPUを採用し，Windows3.1にも対応した。また，OADG仕様のOSとして

DOS/VをROMで搭載している。画面表示は，ハイコントラスト白黒液晶画面で640×480ドット，16階調のVGAモードに対応した。ワープロ機能としては，図形作成や文書デザインを助ける「アート倶楽部」がある。ほかにも「書院スーパーアウトラインフォント」6書体をROMに内蔵した。辞書は日常用語をはじめ，固有名詞など約92万語を装備している。また，コードレス光通信機能により，同社の新携帯情報ツール「PI-3000」(ザウルス)，電子マネージメント手帳「PV-F1」などとデータが双方向で通信可能。

「PC-WD2AD」は，「PC-WD2A」の機能と同等で，120Mバイトのハードディスクを内蔵し，DOS/Vをハードディスク内に搭載している。

価格は，「PC-WD2A」が298,000円，「PC-WD2AD」が378,000円(ともに税別)となっている。

〈問い合わせ先〉

シャープ㈱　☎06(621)1221，043(299)8210

レーザープリンタ
LP-1500S/LP-8000S
エプソン

エプソンは，レーザープリンタ「LP-1500S」「LP-8000S」を発売した。

「LP-1500S」は，600dpi相当の高精度印字となっている。A4用紙で1分間に6枚の印刷が可能。用紙サイズはハガキからA4まで対応している。また，作成したデータは80%の縮小印刷ができる。標準の用紙トレイでは150枚，オプションのロアーカセットにより最大400枚の用紙がセット可能。アウトラインフォントについても，明朝体，ゴシック体を標準で装備し，オプションで正楷書体，行書体(毛筆)などが用意されている。ほかにもオプションのインタフェイスをつけると，2種類のパソコンのデータを自動的に切り替えて印刷を行う。

「LP-8000S」は，「LP-1500S」の機能に加え，プリンタコントローラに32ビットCPUを搭載し，出力時間を従来機に比べて約20%短縮させた(同社調べ)。A4用紙で1分間に8枚の印刷が可能となった。用紙サイズはハガキからA3まで対応。最大250枚までセット可能な用紙カセットを標準装備している。オプションには，ハガキを100枚セッ



LP-1500S



LP-8000S

ト可能なハガキカセット，3種類の用紙を最大500枚セットできるダブルカセットユニットなどがある。印刷の自動切り替えは3種類のパソコンで可能。

価格は，「LP-1500S」が148,000円，「LP-8000S」が298,000円(ともに税別)となっている。

〈問い合わせ先〉

エプソンインフォメーションセンター

☎03(3377)3500，06(212)8712

電子メモ
WN-L10/WN-L20
シャープ

シャープは電子メモ2機種「WN-L10」「WN-L20」を発売した。

本機は，友人，お店などのリストや予定などの個人データを記憶するものである。

「WN-L10」は，電話帳機能が友人リストとお店リストの2つに分かれている。入力項目は名前，電話番号，備考メモの3つ。それぞれの備考メモの部分にキーワードをつけてデータを分類することで，素早い検索が可能。予定の記憶は日付，時間(1種類)，出来事の3つの項目を入力する。日付


WN-L10


WN-L20

の過ぎたデータも消去されないので日記としても利用できる。記憶容量は，名前が8文字，電話番号が12桁の場合，約100人分のデータが保存できる。

「WN-L20」は，電話帳機能が「WN-L10」のように2つに分かれていない。記憶容量は「WN-L10」と同条件の場合，約555人分のデータが保存できる。

また，両機種ともにインディゴブルーとコットンベージュの2色から選べる。

価格は，「WN-L10」が5,000円で「WN-L20」が7,500円(ともに税別)。

〈問い合わせ先〉

シャープ㈱☎06(621)1221，043(299)8210

INFORMATION

チケットプレゼント
GAMADELIC LIVE
DATA EAST

GAMADELICは，1月21日に発売される初のベストアルバムを記念して，ライブを開催する。そのチケットをベストアルバム「DELICIOUS SELECTION」についている応募券を送ってくれた方，先着250名にプレゼントする。当日はライブのほかに，新作ゲーム展示コーナーやCD販売コーナーが予定されている。

ライブの日時は，2月5日(土)の18：30開場，18：30開演。場所は原宿ルイード。

〈問い合わせ先〉

サイトロン・アンド・アート㈱

☎03(3498)7273

FILES Oh!X

このインデックスは，タイトル，注記――著者名，誌名，月号，ページで構成されています。2月に入って，受験生の方にはますます辛い季節になってきましたね。体調を整え最高の結果が出せることをお祈りしています。

参考文献
I/O　工学社
ASCII　アスキー
コンプティーク　角川書店
C Magazine　ソフトバンク
テクノポリス　徳間書店
電撃王　主婦の友社
POPCOM　小学館
マイコンBASIC Magazine　電波新聞社
My Computer Magazine　電波新聞社
LOGIN　アスキー

一般

▶THE NEWS FILE
機能充実のフォトCDプレーヤー，コダック「PCD970」や，3D文字が作れる新型ペン書院など，ハイテク関連グッズを紹介するコーナー。――編集部，LOGIN，24号，32-39pp.

▶3DOが来た!!
10月にアメリカで発売された3DO。早くも6万台を出荷したというこの注目のハードを入手し，その実態をレポートする。――編集部，LOGIN，24号，264-265pp.

▶電網幼稚園
パソコン通信初心者のための教育ページ。最終回となる今回はパソコン通信に役立つフリーソフトウェアの紹介を行う。――編集部，LOGIN，24号，288-289pp.

▶NEWS COLLECTORS
シリーズ「次世代ハードの真実を探る!!」第1回の3DO編など，コンピュータ関連のニュースをまとめて贈る。――編集部，電撃王，1月号，10-13pp.

▶年末年始アーケードゲーム事情
人気格闘ゲームの分析や年末登場予定のゲームレビューを行う。――編集部，電撃王，1月号，61-68/117-129pp.

▶クレーンゲームの謎と真実
女の子をゲームセンターに引き寄せる主役のクレーンゲーム。その実態について，バンプレストにインタビューする。――編集部，電撃王，1月号，88-91pp.

▶特集　コンピュータ・ミュージックに挑戦
MIDI規格の解説から始まり，音源のセッティング，OSやゲーム上での使われ方などを中心に紹介する。コンピュータミュージック関連製品の一覧つき。――編集部，マイコンBASIC Magazine，1月号，41-53pp.

▶Bug太郎のプログラム・タイム
レースゲームにチャレンジ。BASICでトップビュー型のレースゲームをプログラミングしてみる。――谷裕紀彦，マイコンBASIC Magazine，1月号，88-89pp.

▶BASIC MAGAZINE NEWS
ソニーの家庭用ゲームマシンやTAKERUの「名作文庫ソフト」などのニュース。――編集部，マイコンBASIC Magazine，1月号，179-181pp.

▶海外発国産機種行
1993年の年末から1994年の春にかけて，海外生まれのビッグタイトルがぞくぞくと日本に上陸する。その顔ぶれと内容を紹介。――編集部，マイコンBASIC Magazine，1月号，195-207pp.

▶特集　パソコンゲームってなくなるの？
パソコン界の現状を検証し，クリエイターの座談会などを通じてパソコンゲームの未来を予測する。――編集部，コンプティーク，1月号，35-47pp.

▶大特集1　新春おめで対談!!
筋肉少女帯の内田雄一郎VS千葉麗子，「ミッションスクール大図鑑」の森伸之VS「同級生」の藤田正人という対談企画の2本立。――編集部，POPCOM，1月号，6-12pp.

▶NEWS CLIP
Macintoshの新ラインナップ，NECの「デジタルブックプレイヤー」などの新製品紹介や各種ショウのレポート。――編集部，POPCOM，1月号，51-56pp.

▶イチンパ!!
女の子のパソコンライフを考えるページ。今月は，マシンスペースをどう可愛く飾るか。――編集部，POPCOM，1月号，164-165pp.

▶新製品良品館
DCCコンポやポータブルMDなどの音響製品を特集。――編集部，POPCOM，1月号，174-175pp.

▶特集　1994年は絶対にこうなる！
シミュレーションゲーム，ロールプレイングゲームのクリエイターの対談を通じて1994年の傾向を探り出す。また，インタビューによる科学技術全般の予測などもある。――編集部，LOGIN，1・2合併号，147-165pp.

▶THE NEWS FILE
低価格INDYや，デジタルベータカムが登場したニコグラフのレポート，低価格CD-ROMドライブ一挙紹介など，ハイテク関連のトピックを集めたコーナー。――編集部，LOGIN，1・2合併号，166-173pp.

▶Pentiumとはなんだ？
次世代CPUの期待がかかる，そのうちのひとつPentiumを紹介する。ほかにも，インテルのCPUの歴史に触れる。――編集部，LOGIN，1・2合併号，204-207pp.

▶特集　大容量記憶装置
HDD/MOの導入法や使用法をアドバイス。――長谷川博之ほか，I/O，1月号，77-92pp.

▶秋季コムデックス'93
RISCビジネスに意欲的なアップルの基調演説，PDA市場などの刺激的な内容をレポート。――Dana Blankenhorn，I/O，1月号，113-116pp.

▶マルチメディアの行方
Panasonicの3DOマシン「REAL FZ-1」から見る，CD-ROMによるマルチメディアの現状と行方。マルチメディアの企業提携図付き。――奥野雅之，I/O，1月号，135-138pp.

▶ASCII EXPRESS
「COMDEX FALL'93」「マルチメディア'93」の模様をレポート，そのほか新製品情報が満載。――編集部，ASCII，1月号，234-236pp.

▶特集1　'94年最新機種ガイダンス
海外マシン数機種を含む，各社ニューモデルの実力検証。異機種間ベンチマークによる比較つき。――編集部，ASCII，1月号，257-280pp.

▶特集2　情報戦を生き残れ
電子手帳などの情報管理ツールの動向や製品紹介，活用法などを紹介する。――編集部，ASCII，1月号，305-320pp.

▶FORMULA ONE COMPUTING
'93秋の鈴鹿グランプリより，レースの現場で働くさまざまなコンピュータをレポート。――編集部，ASCII，1月号，326-330pp.

▶新科学対話<1>
科学者を迎えての連続対談シリーズ。ゲストは，NHKスペシャル「脳と心」に出演中の養老孟司氏。コンピュータと脳との接点とは。――竹内郁雄，ASCII，1月号，370-376pp.

▶特集　最新FAXモデムと通信ソフト
低価格化，高機能化により確実に普及しつつあるFAXモデム。その活用法を解説するとともに，最新パソコン通信用モデム数機種を紹介。――編集部，My Computer Magazine，1月号，27-42pp.

▶ステップアップに役立つパソコンNEWプロダクツ
周辺機器とパソコン本体，ソフトの新製品情報。――編集部，My Computer Magazine，1月号，66-73pp.

▶特集2　'94新春パソコンマーケットガイド
全国主要パソコンショップの紹介。お得なパソコン購入法や，ショップごとの目玉商品も掲載。――編集部，My Computer Magazine，1月号，171-203pp.

▶ISDN時代のパソコン通信
沖電気の最新モデム「PCLINK TA/296」の紹介を交え，ISDNとはなにかをわかりやすく紹介する。――高橋雄一，My Computer Magazine，1月号，60-63pp.

▶未来派パソコン通信の研究<3>
今月は2，3年先のパソコン通信の未来を予測する。――原田洋平，My Computer Magazine，1月号，138-139pp.

X1/turbo/Z

X1シリーズ

▶BLOCK
ブロックくずし。ラケットが傾くのでボールが思わぬ方向へ飛んで行ってしまうかも。――西尾幸造，マイコンBASIC Magazine，1月号，132-133pp.

▶MARBLE KID
ブロックを自由自在に動かして，モンスターを倒せ！――寺田宏幸，マイコンBASIC Magazine，1月号，134-136pp.

▶Winning Run～SUZUKA～
New FM音源DRIVER上で動作するミュージックデータ。――西尾将人，マイコンBASIC Magazine，1月号，156-157pp.

▶で・ぱ・ぐ
1993年12月号に掲載されなかった「スピンディジーII」の音色設定プログラムリストを掲載。──編集部，マイコンBASIC Magazine，1月号，162p.

X68000

▶最新ゲーム徹底解剖!!
話題の人気ゲームを徹底解剖する。X68000は「ストリートファイターIIダッシュ」。──編集部，LOGIN，24号，184-185pp.

▶X68030新聞
最終回ということで，「X68030新聞社が選んだこの1本」，ズームの「オーバーテイク」を取り上げる。──編集部，LOGIN，24号，268-269pp.

▶今月の電撃王
パソコン，コンシューマなどのゲームを掲載。X68000版は「餓狼伝説2」「ドラゴンバスター」。新作ソフトリリースカレンダーつき。──編集部，電撃王，1月号，15-29pp.

▶SHORTEN・改
パネルを開いて，自分の領地を広げていく陣取りゲーム。X-BASIC用。──非アクション会会長14才，マイコンBASIC Magazine，1月号，137-139pp.

▶WHOLE HEART OCTOPUS
タコ君が活躍するアクションゲーム。──高橋秀之，マイコンBASIC Magazine，1月号，140-142pp.

▶サムライスピリッツ～自然の宴（ナコルル）～
ミュージックプログラム。──藤井亘，マイコンBASIC Magazine，1月号，158-160pp.

▶SUPER SOFT HOT INFORMATION
ビデオゲームアンソロジー第7弾「ドラゴンバスター」や「餓狼伝説2」，パズルゲーム「キーパー」「宝魔ハンターライム第5話」を紹介。──編集部，マイコンBASIC Magazine，1月号，とじ込み付録10p.

▶SUPER SOFT EXPRESS
別冊ソフトガイド。「餓狼伝説2」「卒業」「麻雀クエスト」「ネメシス'90改」などの新作ソフトが紹介されている。機種別の発売予定表もあり。──編集部，コンプティーク，1月号，別冊19-20，26，34pp.

▶How to Win
「項劉記」では人間同士で対戦した場合の勝利の秘訣，「信長の野望・覇王伝」のリプレイなど。──編集部，コンプティーク，1月号，70-73，86-89pp.

▶NEW GAME REPO!!
格闘アクション「ストリートファイターIIダッシュ」を始め，各機種用のパソコンゲームを紹介する。機種別カレンダーつき。──編集部，テクノポリス，1月号，36，47，49，54pp.

▶HOT REVIEW!!
ゲーム業界の有名人たちがゲームソフトを批評するコーナー。X68000版「ぷたさん」が取り上げられているほか，「コットン」の未公開資料も掲載。──編集部，テクノポリス，1月号，64，66pp.

▶DO-JIN SOFT FAN!!
アマチュアソフトを紹介するコーナー。X68000はファンタジックな世界が舞台の「SILK ROAD」，対戦プレイがアツい格闘ゲーム「Power Knucle」などが登場。コミケットサークル予定表もある。──編集部，テクノポリス，1月号，70-78pp.

▶新作パーフェクトガイド
X68000は「餓狼伝説2」「ドラゴンバスター」を紹介。──編集部，POPCOM，1月号，45，49pp.

▶68新聞
新装開店で登場の，X68000情報ページ。「餓狼伝説2」「ストリートファイターIIダッシュ」「ドラゴンバスター」を取り上げる。──編集部，LOGIN，1・2合併号，186-187pp.

▶FREE SOFTWARE INDEX
大手主要ネットにアップロードされたソフトを紹介。X68000は，SX-WINDOW上で6万色背景を実現する「grroot.x」，SX-WINDOW対応の麻雀牌を使ったパズルゲーム「SX青海.X」のフリーソフト情報が紹介されている。──編集部，ASCII，1月号，457p.

▶HOBBY EXPRESS
マニア必携ゲーム「ぶたさん」の紹介。──編集部，My Computer Magazine，1月号，206p.

▶なんでもQ&A
「Easydraw SX-68K」を使い，網かけパターンの中に中抜き文字を重ねる方法ほか，テキストプレーンの消去法など。──編集部，My Computer Magazine，1月号，160-161pp.

▶SX-WINDOWプログラミング
今回は基本的な操作のひとつ，ウィンドウを開くことに挑戦する。──吉野智興，C Magazine，1月号，140-146pp.

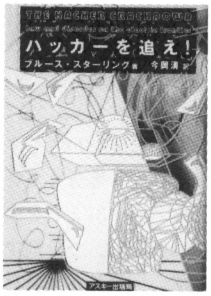
ポケコン

PC-E500

▶アンド君の帰宅Version 2.2
暗くなるまでに家に帰らないと……。タイミングが命のワンキーゲーム。──西野陽一，マイコンBASIC Magazine，1月号，145-146pp.

▶DOT DRAGON
せまってくるブロックをよけるアクションゲーム。──たりらりらん，マイコンBASIC Magazine，1月号，147-149pp.

新刊書案内



ハッカーを追え！
ブルース・スターリング著
今岡清訳
アスキー刊
☎03(5351)8194
四六判　448ページ
2,980円（税込）

アメリカのハッカーネタ本というのはいろいろと出ているわけだが，今回のように著者がブルース・スターリング（サイバーパンクSF作家）で，訳者が今岡清(元SFマガジン編集長)，装丁が立花ハジメとくれば，つい手にとってしまうのが人情というもの。ブルース・スターリングであるから，ドキュメンタリーではあるが，いままでのハッカーものとはひと味違う。システム破りをする系統のハッカー(クラッカーと呼んだほうがよいか？）たちと，彼らの世界に関するクールな物語なのだ。決して，世間一般にハッキングの世界を教えるための啓蒙書でも，セキュリティ強化を勧めるための本でもない。現実世界とサイバースペースの戦いがテーマであり，ハッカーと電話フリークのアンダーグラウンドで活躍する人たちをクールに描くのがテーマなのだ。まさに，現代のサイバースペースドキュメンタリーなのである。

キーになるのは，1990年の5月8日に行われた，シークレットサービスのハッカー一斉取り締まりだ。なかでも「サンデヴィル作戦」では，40台のパソコンと23,000枚のFDなどが押収された。著者は決してハッカーや電話フリークを擁護しないが，現実世界（特に政治的権力）がサイバースペースに現実的な対応を持ち込むことをよしとはしない。著者は本書の中でこう書いている。「ぼくには，アメリカの作家たちは封をされた捜査令状によって，なんの罪に問われることもないままコンピュータを押収されるかもしれない，ということがわかり始めてきた。……（中略）……いまや電子的な表現の自由と，コンピュータ犯罪の現実だとされている世界に入っていく時だった。その結果がこの本だ。その結果が電話会社の世界であり，デジタルアンダーグラウンドなのだ。」

サイバースペース黎明期の記録として，いままでの視点では見えなかったデジタルアンダーグラウンドを知る好著として読んでみたい。（K）



SFアニメを天文する
福江純著
日本評論社刊
☎03(3987)8611
新書判　152ページ
1,500円（税込）

鉄腕アトム，機動戦士ガンダム，風の谷のナウシカ，この3本のアニメを読者のほとんどの人が知っているであろう。では，機動戦士ガンダムのなかで登場したスペースコロニーを現実に建設するならばどこにすればよいか？　この問いに理論的に答えられる人がどれだけいるだろうか？

30年来アニメとつき合ってきた著者が，いろいろなアニメを題材として天文学に関係する基本的なことや最先端の話を丁寧に解説している。本書を読めば，これまで見たアニメでなんとなく見過ごしていたことへの理解が深まり，そのアニメをより楽しめるようになるだろう。



ビジュアル
ディクショナリー1
日用品
EVERYDAY THINGS
徳永優子翻訳
同朋舎出版刊
☎075(212)5900
B4変形判　69ページ
2,800円（税込）

いま，みなさんの日常で使っているものがどんな部品からできているか知っているだろうか？たとえば，自転車，時計，傘。それぞれ自分で使ってはいるけれど，「どんな部品で，いくつの部品で作られてるの？」と急に聞かれてもわからない。でも，この本を見ればそれがわかるのだ。

本書では，日用品が部品ごとに分解された状態で写真やイラストに収められている。そして部品ごとの名称，その英訳が紹介されている。もちろん，部品の名称がわかれば，索引でその形状が調べられる。これはシリーズもので，ほかにも「人体」「船と航海」「動物」「植物」などがある。

QUESTION and ANSWER

Q MUSIC PRO-68Kで作成されたMMLデータの曲をアセンブラで演奏するにはどうしたらいいのか教えてください。 愛知県 小田島 倫也

A MUSIC PRO-68Kで作成したデータのMML化を行うと＊.MUSという形式になります。これは＊.OPMによく似た形式のテキストファイルですので，簡単な処理で＊.OPMファイルに変換することができます。

とりあえず，X-BASICでプログラムを組むなどの方法でも処理できますが，Z-MUSICシステムをお使いならば，もっと簡単に処理することができます（バージョンは問わず）。

まず，BASIC.CNFにMUSICZ.FNCを組み込んだうえで，X-BASICを立ち上げ，

M_SWITCH(1)

と入力します。

次にMUSIC PRO-68Kに付属していたPL.BASを使用してデータを演奏してください。自動的に“ZMUSIC.ZMS”というファイル（＊.ZMSファイルは＊.OPMファイルとほぼ同等なもの）を作成してくれます。

このようにしていったん＊.OPMファイル形式に変換してしまえば，あとはそんなにややこしくありません。

考え方としてもっとも単純なのは，そのファイルを“OPM”というファイル名で書き込んでやることです。これならばファイル操作だけですみます。メモリ上にMMLデータを保持している場合には，Z-MUSICを使うならば，

ZMUSIC-C TEST.OPM

のようにデータをZMD化したものをメモリ上に取り込んでおき，ファンクション11_H(play_conv_data)でより高速に演奏することができます。

D1.L=11_H
D2.L=0
A1.L=データ格納アドレス＋7

のようにレジスタを設定し，

TRAP #3

を実行してください。

Q 256色モード用のグラフィックエディタを65536色モードで作りたいと思っています。256色モードで使用されているパレットがよくわかりません。緑3ビット，赤3ビット，青2ビットで色を作ってもなぜか違ったものになります。デフォルトではどのような色が割り当てられているのでしょうか。 大阪府 高宮 慎一

A おそらく少し考え違いをしているのだと思います。たとえば青であれば0～31の区間を4等分すればいいのですから……と，32÷4＝8になるので，

```
FOR I=0 TO 3
    B=I*8
      :
NEXT
```

といった内容の処理をしてしまいがちです。しかしこれでは0～24までを4等分していることにしかなりません。0～31までを4等分するには，32÷3＝10.666...で分割すればよいことになります。同様に0～31を8等分するには，32÷7＝4.57...で分割すればいいことになります。

きちんと割り切れないので誤差が出る可能性はありますが，このような処理でシステムが扱っている256色モードのデフォルトパレットと同等なものが作成できます。

Q Oh!X1993年11月号80ページに「SX-WINDOWではツァイト社から出ている和文アウトラインフォントを使用することができる」とあり，その下にはJGフォントの印字見本もあります。一方，SX-WINDOW ver.3.0ユーザーズマニュアルの163ページでは，フォントマネージャで対応しているのは「拡張子がIFMのものと書体倶楽部のもの」とあります。おそらくフォントマネージャに手を加えなければならないのだと思いますが，JGフォントのシャーペンなどでの使用方法を教えてください。 東京都 佐川正人

A ひと言でいうと特別な操作はなにも必要ありません。SX-WINDOW ver.3.0のシステムに付属するIFM.ENVというファイルの内容を表示してみてください。すでに主なJGフォントが登録されているはずです。ですから，買ってきたJGフォントファイルをIFMで指定されたディレクトリに放り込めばすぐに使用できるようになります。

ですから，JGフォントをSX-WINDOWで使用するうえで注意しなければならないことといったら，元ファイルがすでにベジエ曲線のデータなのですから無駄なベジエ化などをしないようにすることぐらいでしょうか。そのほか，FLOAT4.X（X68030の数値演算コプロセッサ用浮動小数点演算ドライバ）を使用していると，ごくまれにアウトラインフォントの展開に失敗することがあるようですが，これもFLOAT2.Xならば大丈夫なようです。安心して使用してください。

（中野 修一）

Q Z-MUSICシステムver.2.0を買いました。しかし，ドキュメントで書かれているBOS.CNFというファイルがみつかりません。 埼玉県 山本博晃

A ようやく発売されたZ-MUSICですが，手違いによりいくつかのファイルが抜け落ちていました。

とりあえず，これまでのデータを演奏するために必要なものを掲載しておきます。まず，OPMDRVn.X用のデータを再生するために必要な“68SND.ZMS”，そして質問にもある“BOS.CNF”です（OPMDのデータを再現するために必要）。

“68SND.ZMS”は要するにOPMDRVn.Xの内蔵音色ですから，OPMDRVn.Xから自動生成します。OPMDRV.X，OPMDRV2.X，OPMDRV3.Xのいずれかを組み込み，MUSIC.FNC（MUSICZ.FNCは不可）を組み込んだX-BASICを立ち上げ，リスト1のプログラムを入力して実行してください。これで自動的にファイルができあがります。

“BOS.CNF”はエディタからリスト2をそのまま打ち込んで使用してください。Z-MUSICシステムver.1のときのものでもかまいませんが，内容は一新されていますのでできるだけ新しいものを使ったほうがいいでしょう。

また，サンプル曲のうち，

CT2.ZMS　　：作曲 矢部雅敏
SLOPE.ZMS　：作曲 矢部雅敏

のAD PCMパートが正常に演奏されませんでした。DISK 1のAUTOEXEC.BAT中の，

if exitcode 2 set zmusic =%zmusic%;B:¥BASS;B:¥SNARE;B:¥TOMTOM;B:¥ETHNIC;‥‥

の最後尾に，

;B:¥

を追加してください。

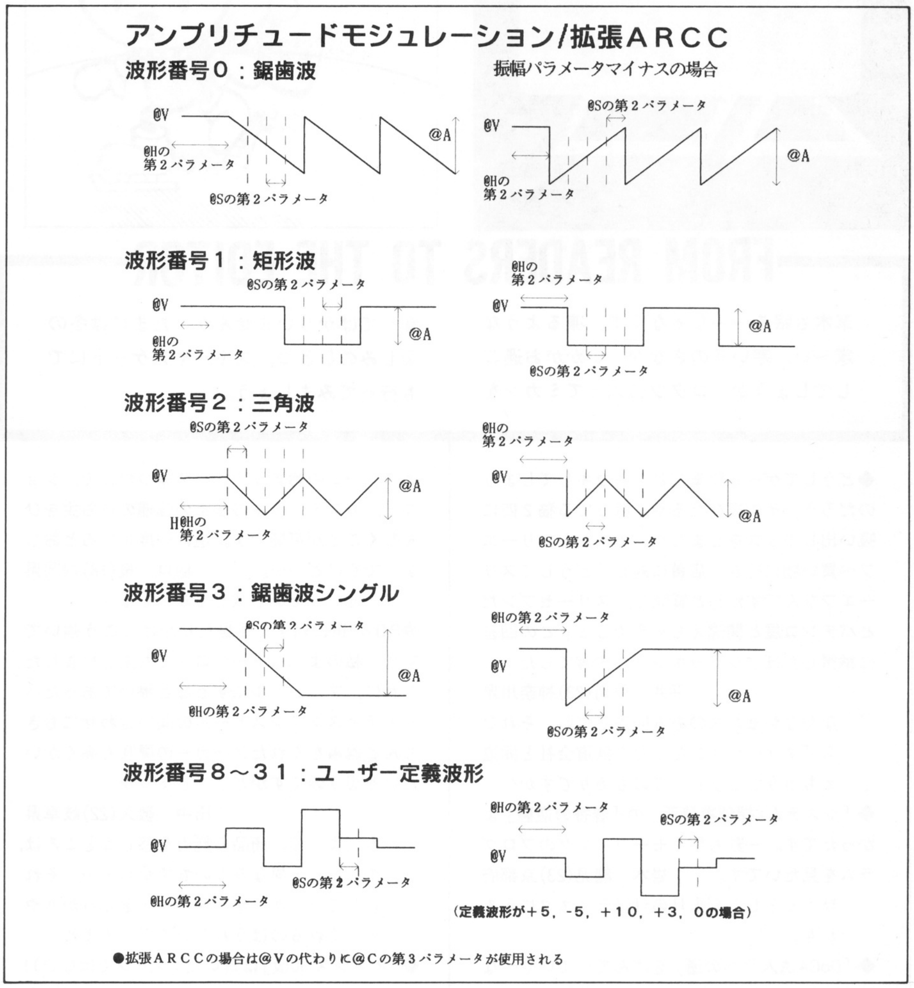
また，42ページの図に誤りがありました。正しくは右の図のようになります。70%に縮小して上から貼り込んでください。

さらに，ZPCNVで標準ドラムセットのうちのTR-808セットを作成しようとすると，TR808HO.PCMの処理中に“CUT SIZE IS TOO BIG.”のエラーが発生します。エラーが発生する行（11行目）にある“C”オプション指定をまるごと削除してください。つまり，正しくは，

.o2a＋＝tr808set\tr808ho.pcm,f3000,0,v94

のようになります。

今回掲載できなかったファイルについては次回の付録ディスクで収録する予定です。ご了承ください。（U）



リスト1

```
10 str s[255],cr
20 char m(4,10)
30 int f
40 cr=chr$(13)+chr$(10)
50 f=fopen("68snd.zms","c")
60 for i=1 to 64
70 m_vget(i,m)
80 fwrites("(v"+str$(i)+",0",f)
90 for j=0 to 4
100 s=cr
110 for k=0 to 10
120 s=s+","+right$("  "+str$(m(j,k)),3)
130 next
140 fwrites(s,f)
150 next
160 fwrites(")"+cr+cr,f)
170 next
180 fclose(f)
```

リスト2　BOS.CNF

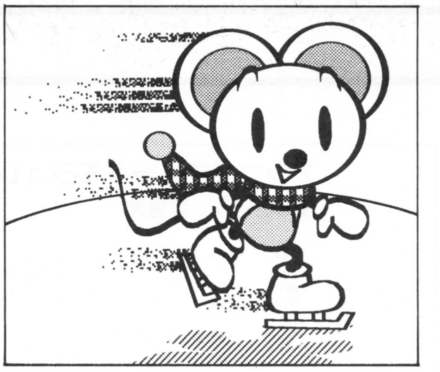
```
01 = whip1.pcm
02 = bass1.pcm,v80
03 = crash.pcm,v40
04 = cup.pcm,v50
05 = wds1_.pcm,m03,p-1,v60
06 = ch1.pcm,v60
07 = oh1.pcm,v60
08 = clp.pcm,v50
09 = wdsd1.pcm,v60,p-1,m02
10 = wdsd1.pcm,v50,p-1
11 = tom5.pcm,v140
12 = tom6.pcm
13 = tom7.pcm
14 = bos_sn.pcm,v120
15 = bos_sn.pcm,v90
16 = bos_sn.pcm,v50
17 = snare3_.pcm,p-2
18 = snare3_.pcm,v80,p-2
19 = snare3_.pcm,v50,p-2
20 = snare4_.pcm,p1
21 = snare4_.pcm,v80,p1
22 = snare4_.pcm,v50,p1
23 = kick3.pcm,v80
24 = hlt4.pcm,v80
25 = hlt3.pcm,v80
26 = hlt2.pcm,v80
27 = hlt1.pcm,v80
28 = etomc.pcm,v50
29 = etoma.pcm,v50
30 = etomg.pcm,v50
31 = etomf.pcm,v50
32 = timbh.pcm
33 = timbl.pcm
34 = lowtb.pcm
35 = 34,p-3
36 = oh2c.pcm,p-5,v90
37 = oh2c.pcm,p-4,v90
38 = oh2c.pcm,p-3,v90
39 = oh2c.pcm,p-2,v90
40 = oh2c.pcm,p-1,v90
41 = oh2c.pcm,v90
42 = oh2c#.pcm,v90
43 = oh2d.pcm,v90
44 = oh2d#.pcm,v90
45 = oh2e.pcm,v90
46 = oh2f.pcm,v90
47 = oh2f#.pcm,v90
48 = oh2g.pcm,v90
49 = oh2g#.pcm,v90
50 = oh2a.pcm,v90
51 = oh2a#.pcm,v90
52 = oh2b.pcm,v90
53 = oh2cc.pcm,v90
54 = hlt4.pcm,v60
55 = hlt3.pcm,v60
56 = hlt2.pcm,v60
57 = hlt1.pcm,v60
58 = 28,v80
59 = 29,v80
60 = 30,v80
61 = 31,v80
62 = 11,v70
63 = 12,v70
64 = 13,v70
65 = ch1.pcm
66 = oh1.pcm
```

質問にお答えします

日ごろ疑問に思っていること，どんなことでも結構です。どんどんお便りください。難問，奇問，編集室が総力を挙げてお答えいたします。ただし，お寄せいただいているものの中には，マニュアルを読めばすぐに解答が得られるようなものも多々あります。最低限，マニュアルは熟読しておきましょう。質問はなるべく具体的に機種名，システム構成，必要なら図も入れてこと細かに書いてください。また，返信用切手同封の質問をよく受けますが，原則として，質問には本誌上でお答えすることになっていますのでご了承ください。なお，質問の内容について，直接問い合わせることもありますので電話番号も明記してください。

宛先：〒103　東京都中央区日本橋浜町
3-42-3
ソフトバンク株式会社出版部
Oh!X編集部「Oh!X質問箱」係

STUDIO X ON AIR



FROM READERS TO THE EDITOR

草木も眠る……じゃなくて，凍るような寒～い，寒い冬のさなか，いかがお過ごしでしょうか。コタツに入ってミカンを食べてばかりいませんか。たまには冬の楽しみのひとつ，スキーやスケートにでも行ってみましょう。

◆どうしてゲーム特集だとウキウキしてしまうのだろう。その喜びたるや，飼ってる猫2匹に吸い出しキッスをしました。それからスリーエフへ買い物に行き，店員に対し「どうしてスリーエフなんですか?」と質問し，「スリーセブンだとパチンコ屋と間違えちゃうでしょ」との回答に納得したほどショッキングな内容でした。

平井　秀司(23)神奈川県

なかなかセンスのある回答ですね。それなら「スリーナインだったら鉄道会社と間違えちゃうでしょ」ってのもありですか?

◆「システムX探偵事務所」の「春香の逆襲」よかったです。一刻も早くモーフィングのプログラムを見たいです。　岩本　龍児(23)京都府

ひょっとしたら来月の付録ディスクに入るかも。

◆「DōGA法人化への道」を読んでいていや～な気分になってきた。おそらく日本全国がこうなのだろうが……。規制緩和は名ばかりか?　ほかの「文化」にかかわる公益法人はどうやってやるんだろう?　村上　智一(18)熊本県

そういえば規制緩和が行われるどころか，規制は増えているという記事を雑誌で見たような……。

◆「善バビ」にも（毎回の）タイトルがあったほうがいいな。INDEXがとっても寂しいから……。というのは余談として，INDEXには毎回感謝しています。調べものがすぐに見つかりますから。ほかの雑誌だと下手をすれば2～3年分ひっくり返すときもありますから……。

北口　修一(20)兵庫県

そういっていただけると，とってもありがたいです。こちらの苦労も報われます。みなさん参考にしてくださいね。

◆「スーパーリアル麻雀PII&PIII」はめっきり弱くなっています。ところが，ゲーセンではPIVがあいかわらず，多くの若者を地獄の淵まで超特急で案内しています。STARTするときにカン・ポン同時押しで2人目，チー・リーチ同時押しで3人目から始められます。3人目に勝つとボーナスゲームの始まりで，一度勝つだけで，ショウ子，カスミ，ミキの各々の強運の持ち主をひんむくことが可能です。逆に一度負けるとおしまいですけど……。　樋口　泉(18)福岡県

今度ゲーセンで試してみましょう。

◆PII&PIIIに対する風当たりがけっこう強いですが，私のようなトーシローでも楽しめました。それに，マニュアルにするなと書いてあったハードディスクインストールの問い合わせにもきちんと返事をくれたメーカーの厚意も高くかいたいと思うのですがどうでしょうか。

田中　敬久(22)岐阜県

人によって，商品の魅力を感じるところは，ずいぶんと異なるものですから……。それにしても，ユーザーサポートをしっかりやってくれるのはうれしいことですよね。

◆「ネメシス'90改」は買いだ（シャレではない）!　けれどMSX版に比べてランクの上がり方が尋常でない気がする。せめて「ガリウスの迷宮」さえあれば……。裏技を誰か見つけてほしい。

立花　弘史(20)熊本県

楽しんでいるみたいですね。「ガリウスの迷宮」が見つかるといいのですが……。

◆「GCCによるX680x0ゲームプログラミング」はなんだ!　GCCを7,000円，ライブラリを9,000円（だったっけ?）で買った私は愚か者だったのか。　藤田　敬(26)宮崎県

Cにより早く親しめたのだし，マニュアルもしっかりしたものが手に入ったのだから，愚か者なんてことはありませんよ。

◆とうとう買いました，「ストリートファイターIIダッシュ」。X680x0で出るんだったらメガドライブ版なんて買わなかったのに……。CPSファイターアダプタ4,000円するんですか。メガドライブのコントローラを使うんだったら，このアダプタを買うよりも「チェルノブ」を買ったほうが安いですね。　北浦　暁光(19)東京都

実際に「チェルノブ」の変換コネクタはキーの割り当てをコンフィグで替えないといけないけど，使えますよ。

◆最近SLGがおもしろくないと思いませんか?　僕はPC-9801，FM TOWNSを経てX68000へきた者ですが，結局ACTがいちばん効率のよいおもしろさを運んでくることに気づきました。別にSLGが下手というわけではないのですが……。

藤原　正浩(16)福岡県

ちょっと毛色は変わってますけど「ロボットコンストラクションR.C.」なんかおもしろいと思いますけど。

◆5年前，「ドラゴンスピリット」をしたくてX68000を購入……。月日は流れ手持ちのゲームソフトもたくさんたまってしまった。いい歳をしてと思われるかもしれないが現役だ!　といいたい。もちろん仕事にも使っていますよ。

尾形　淳一(40)北海道

いくつになっても楽しいものは楽しいですよね。仕事にはどんなソフトを使っているんですか?

◆Oh!Xを1～12月号まで初めて買い続けました。そして1993年はいろんなことがありました。1月号，初めてハガキを書く。2月号，受験が忙しくなる。3月号，X68000もこれまでか……。4月号，プレゼントに当選する。5月号，ちゃだワを読む。6～9月号，プレゼントがこない。10月号，初めてのディスク。11月号，X68000XVIを持つ友人発見。12月号，プリンタが壊れる。そしてプレゼントがまだこない。



◀岡村　直也　兵庫県

黙ってもっていかれようとしているゲームソフトをほっといてもいいのかな?　残りの2話も楽しみですが、その次も期待していますよ。



◀小川　伸輔　宮城県

小川さんも春には幸せが訪れるといいですね。ありふれてますが、「サクラサク」ってな感じですか?

片山　明義(15)奈良県

発送が遅れて申し訳ありません。今月号が出るころには届いていると思うのですが。

◆パソコンを買った当初，X68000のことを書いた本が見つからなくてシャープのパソコンを取り上げる雑誌はないと思っていました。そんなとき上司にOh!Xを教えてもらい，ずっと参考にさせてもらっています。Oh!Xを読んでいてつくづく感じたのはユーザーあってのメーカーだということとユーザーがX68000を進化させているということです。　細谷　栄一(29)奈良県

いい上司ですね。これからもよろしくお願いします。

◆X68030が発売されて半年たったので，そろそろOh!Xの表紙にあるand X68000をand X680x0にチャレンジしませんか? X68030ユーザーにとって，とても気になることなんです。

須田　周作(21)岡山県

いろんなことが気になるんですね。今度相談してみましょう。

◆某月某日，FM R用「PITMAN」をプレイする。……泣ける。この正月に田舎に帰ったら久しぶりにMZ-700を起動してみよう……。

杉本　秀昭(23)神奈川県

たまには懐かしいパソコンを起動してみるのも楽しいかもしれませんね。

◆先日，いきなり停電になった。愛機のX68000 XVIは主電源を切っており，当然ハードディスクは動いていない。よかった。

河本　直規(28)和歌山県

とりあえずよかったですね。でも，いつも主電源が入ってないわけではないですよね。

◆以前，ダンプリストを入力しているところを女に見られた。いまはもういない。たまにダンプリストを入力していると切なくなる。声に出して打ち込んでたのがアレだな。いまは黙って打ち込んでいる。　田中　信一(22)神奈川県

まだまだ寒い日が続きますが，がんばって春を迎えましょう。

◆ここ数年，パソコン業界はパッとしませんね。そういえば僕が中学生のときも同じようなことを思っていました。そのときは，X68000というまったく新しいコンセプトと圧倒的なスペックをもったマシンの登場で業界も沸きましたし，僕も目からウロコが落ちました。もうそろそろあのときのようなショックをもう一度味わいたいと思うのは僕だけではないはずです。

堤　哲也(21)福岡県

そんなショックだったら一度といわず何度でも味わいたいですね。

◆帰省のときに使うためMacintoshを買いました。使っていくうちにX68000にはない魅力，またX68000にしかない魅力が見えてきました。いままで井の中の蛙だったことを思い知らされました。　信太　徹(23)神奈川県

PC-9801，DOS/Vなどを経験すれば，もっといろんな世界が見られるかもしれませんね。

◆友人2人をX68000にハメてしまいました。しかも2人ともX68000を買っちゃって，私ってイケナイことをしたのでしょうか?

河上　博仁(17)埼玉県

ハメたなんてとんでもない。素晴らしい世界に案内したんですから，これからも面倒をみてあげてくださいね。

◆先日，あの奥尻島行きのフェリー乗り場が目の前に見える父の実家へ行った。そこの町内のゲームコーナーで，往年のテーブル筐体のなかに向かいどうしに並んだ「スパII」の対戦台を発見。うーむこれならお互いにニラミをきかせて対戦できて楽しそう(笑)。しかし相手がもしきれいなお姉さんだったら……。ちなみにこのゲームコーナーは某協同組合の屋上でした。

堀井　晶司(21)北海道

負けた悔しさでいきなり向かいの人に殴りかかって本当のストリートファイトになったりして……。

◆2カ月前からつき合い始めた彼氏の影響でパソコンを始めました。いまは「MATIER」に夢中です。これからはプログラムの勉強したいです。初心者にも楽しめるページがあったらいいなあと思います。　吉野　邦子(23)山口県

今月の特集はいかがでしたか。プログラムを作るきっかけにでもなればいいのですが……。

◆宝くじが当たったー。これを機にX68030を買うぞ! 就職も決まったし，ローンにすればなんとかなるぞ。これでクロックアップしたのにバグりまくりのX68000ACE-HDとおさらばだ。だって一部のゲームにしか対応しないんだもん。

及川　大之(18)東京都

いまごろはX68030で遊びまくっているんでしょうね。それにしてもみんなよく宝くじに当たりますね。少し買ってみようかな。

◆寮に入った（国立の高専にはだいたいあるらしい)。当然，X68000ももってきたのですが，毎晩「餓狼伝説」がアツすぎる。おかげで，通学時間が2時間から5分になったのにプログラムを組む時間ができないし，眠れない日も増えた。

村上　洋樹(17)東京都

いまごろは「餓狼伝説2」でアツくなってるんでしょうね。でも，プログラミングもがんばってくださいね。

◆「GCCによるX680x0ゲームプログラミング」発刊か! よし今度こそCを勉強するぞ，と意気込んで本屋に行ったけど置いてなく，あっという間に「王家の紋章」10冊に化けてしまった。Cマスターへの道は遠い。　安藤　道子(21)宮崎県

きっと，いまは本を手に入れてCマスターへの道を歩んでいますよね。

◆昔，親バカになって子供のビデオにタイトルやテロップを入れてる姿を思いうかべながら初代X68000を手に入れた。ゲームの甘いささやきにそそのかされ回り道をしながら，2本目が完成した。　伊藤　義幸(35)愛知県

X68000を使って撮ったビデオを家族で見ている姿が目にうかぶようです。

◆父がX68030を買ったので，X68000をおさがりとしてもらいました。わーい，これでゲームができるゾー。ワープロもしたいなあ。

安井　久美子(16)京都府

女性のユーザーが増えるのはとってもうれしい。も，もちろん男性のユーザーが増えるのも……。

◆金と銀のエンゼルは，出荷のときの1パック(20個くらい?)に必ずひとつ入っています。見分け方は，まずなるべくたくさんのチョコボールを上から眺められるようにします。次に上からくちばしの部分を見て，最も色の濃い（黄色）ものが当たりです。ここで注意してほしいのは，あまりじっと見つめずぱっと判断することと，当たりがすでにないかもしれないということです。これで何回も当てています。ちなみに大きいチョコボールだと大きい缶詰が送られてきて圧巻です。それと最近はエンゼルではなくキョロちゃんです。　椎名　美臣(23)東京都

◆チョコボールの銀のエンゼルはケースのなかのいちばん前かいちばん後ろのものに当たりがよく入っています。これで5枚当てました。金のエンゼルは残念ながら見たことがありません。うわさではキャラメルでしか当たらないと聞いたことがあります。　竹原　久(23)東京都

チョコボールについての2通のハガキ。みんなはどちらの方法で当たりに挑戦してみますか。



▲今井　健生　奈良県
「ストII」のイラストはたくさん来ますが，ガイルは初めてだったような気がするので，思わず載せてしまいました。このあたりが狙い目かな。



▲占部　哲彦　広島県
6ボタンパッドがネックで，「ストIIダッシュ」を買ってないみたいですね。ここはパッドコレクターになる気で買ってみてはいかがですか。

◆北海道でも「たほいや」は1993年の4月からやってました。最後の3回ぐらいしか見れなかったけど。そんなことで，いま学校で「たほいや」をやっています。放課後に友人5，6を人集めて図書室から広辞苑を借りて，誰もいない教室で黒魔術のごとくやっています。ちなみに「たほいや」は本屋で売っています。知らない人はぜひやってみてください。はまります。

瀬口　武史(16)北海道

「たほいや」とは言葉の遊びです。親が広辞苑のなかからひとつの単語を選び，子がその言葉の意味をひとつ考えます。それから親が子の答えと自分の答え(正しい意味)をランダムに読み上げます。その答えのなかから，子が自分の持ち点を賭けて正しい意味を当てます。テレビでは，賭け点の最高は3点，正解したときは賭けた点を親から貰えます。子が間違ったときは，親に1点，選んだ答えを書いた子に自分の賭けた点を払います。全員が答えを当てられなかったときは親は2点ずつ貰えます。いかにも広辞苑に載っていそうな答えを考えるのがポイントです。それでは興味のある人は試してみてください。

◆A：「最近くしゃみが多くて……上司にカゼうつされたかな?」私：「最近抜け毛が多くて……上司にハゲうつされたかな?」A：「ふぁ～，今日は寝不足で気分が悪い」私：「あ～，今日は眠くて機嫌が悪い」……。その後，私の肩にポンポンと手が……。ああ運命やいかに。

伊藤　直也(23)静岡県

表側にちゃんと職業が書いてあるから大丈夫だったんですね。よかった。

◆表紙のCGのラジオはさりげなく「OH!X」になってませんか?　杉浦　聡(21)静岡県

いわれてみれば，たしかにそうですね。全然気がつきませんでした。

◆「物書きがコンピュータに出会うとき」(実出直人著，河出書房新社刊，1990年)という本を読んだのですが，非常に面白かったので紹介します。この本の対象とする読者は「論理的な文章が書けないで困っている人文学系の大学院生」というとても特殊なものです。ただ，調査資料の収集，整理，分析から，体系的な構成をもつ論文の下書きから清書まで，自分の思考の補佐をさせながらトータルにコンピュータを用いていきたいと思っているあなたにお勧めします。　小川　知幸(23)宮城県

卒論に困っている人は一度読まれては……。あ，ごめんなさい。もう間に合いませんでしたか?　じゃあ，来年度に備えて読んでみては……。

◆11月19日，Oh!Xの発売日，家の近くの2軒の本屋に行くが「まだ入荷してません」といわれあきらめる。11月22日，別の本屋に電話，「担当の者がいないので明日電話してください」……あきらめる。11月23日，その本屋へ行く。「3冊19日に入荷して売り切れ」あきらめる……わけがない。取り寄せてもらうことにする。11月24日，PM7:30本屋に到着，「PM8:00ごろに入荷」といわれほかの本を立ち読みしていると本棚にOh!Xが……おいおい。　白石　和大(17)山口県

本を手に入れられるまでずいぶん苦労されたんですね。どうもありがとうございます。よろしければ定期購読など……。

◆ふふふ，X68030を注文してしまった。ふふふ。　梶原　修(23)福岡県

早く届くといいですね。

◆最近，つまらないことでイライラしている自分がいやになり，さらにイライラしてしまうという状態になる。やはりCaが不足しているのだろうか。　吉田　務(21)大阪府

では，これからコンビニに行って，ザ・カルシウムとカルシウムパーラーでも買いにいってきますね。

◆「ニケ」という名を聞いて，エリート，金髪，美人で権力者のお姉さんを連想してしまう人はOh!Xには何人いるんだろう。あと，「クロービス」と聞いて全身緑色でハネの生えている女の子を連想してしまう人もいるかどうか，気になるなあ。　林　大助(18)神奈川県

さて，なんのことやらさっぱりわかりません。だから教えてくださいね。

◆クリスマスシーズンを迎え，手品の出張サービスの予約の受け付けを始めました。Oh!Xの編集部へなら格安で参上いたしますので，ご利用ください。　北川　亮(23)東京都

残念ながらクリスマスは過ぎてしまいましたが，次の機会には頼んでみたいですね。また連絡をお願いします。

◆結婚した。X68000より私の相手をして，と妻はうるさい。うれしいような，悲しいような……。「ロボットコンストラクションR.C.」やりたいのに……。　藤岡　孝史(25)神奈川県

のろけにしか聞こえないんですけど……。悔しいから「ロボットコンストラクションR.C.」でも……。

◆所沢航空発祥記念館のエクスペリエンスシアターの「天までとどけ」は一見の価値あり。600円で見られる。音楽がいいですよ。

中林　亮(25)奈良県

地方のいろんなところで面白いことをやってたりしますよね。そういえば，新潟へ旅行に行ったときにも科学博物館（だったかな?）が楽しかったですよ。

◆ナタデココを食べましたか?　おいしいともまずいとも思えない。ミョーな味がしました。このブームはいつまで続くのでしょうか?

伊南　尚幸(18)青森県

これを書いているころには，ポストナタデココといわれるパンナコッタが評判ですが，さて本が出るころはどうなっていますか?

◆先日，ゼミ旅行に行った。山口，津和野，萩だ。そのとき気づいたのだが，やたらと向こうの自販機に「ダイエットペプシ」が目立った。「コカコーラ」の影が薄かったような……。最終日に山に登ったが，足を滑らせて腰を打った。ダメージ80%……。トホホ。

坂井　大吾(22)兵庫県

ダメージ80%ならこれからですよ。そろそろゲージが赤く光って超必殺技が……。それは「餓狼伝説2」でした。失礼しました。

◆卒論でカゼひいた。カゼ薬飲んだら気持ち悪くなった。車に乗ったら免停になった（スピードで……）。その翌日，ハードディスクも……。不幸や……まったく。　宇田　淳一(19)東京都

これだけ不幸が続いたら，あとは幸せがまって……いるかなあ?　まあ，お酒でも飲んで……あ，19歳でしたね。飲んじゃだめですよ。

◆子供はどうしてうなじの毛が長いのですか?

長谷部　一也(19)東京都

長いほうが可愛いじゃないですか。理由になってませんけど……。

◆12月号の岩瀬さんへ，「ベラボーマン」は，まだいいほうだと思いますよ。アカペラで「ムーンライト伝説」とか「乙女のポリシー」なんかを歌うよりかは……。それもカラオケボックスではなく，コンパの会場で一般人も周りにいたし，聴いているほうが恥ずかしくなってしまいました。もうしないでくださいね。某大学アニメ研の部長さん。　津村　忠蔵(18)佐賀県

若いっていいですよね……?

◆大学の実験が終わって帰宅する途中である(PM11:30)。しんどい。片道2時間はつらい。



◀岩瀬　貴代美　福岡県

自分の卒業とイメージをだぶらせたりしてるんでしょうか。あ、ゲームは高校生だけど、岩瀬さんは大学生でしたね。ごめんなさ～い。



◀武田　正道　兵庫県

やっぱり冬はスキーですよね。お金がかかるけど。でも淡路島にはスキー場がないんですね。淡路島よりも南の高知県にはあるんですけどね。

小野　貴司(21)奈良県
帰宅できるだけいいかもしれませんよ。今日もまた……。

◆この時期はもう卒業に向けて忙しい。ああ忙しい，忙しい。　松本　太(23)滋賀県
こちらも周囲の人を見るとみんな忙しいみたいです。来月号はディスクもつくし……。

◆パチンコを打ちに行った。隣に学ランを着ている奴がいるにもかかわらず，私だけ年齢を聞かれた。なにやってんだ店員！　ちなみに私は童顔だといわれたことはありません。
内田　大輔(19)東京都
店員さんは，学ランを着たお兄さんがよっぽど怖かったんじゃないでしょうか。だから隣の内田さんに年齢を聞いて，学ランのお兄さんにプレッシャーをかけたかったのでは……。

◆埼玉の三浦貴至さん，北海道でレッズを応援しています。来年こそは必ず勝ってくれますよ。だから川淵チェアマン，柏レイソルと入れ換え戦なんていわないでくださいね。柏レイソルファンの方ごめんなさい。　渡辺　圭(20)北海道
柏レイソルもJリーグで見たかったですよね。ちょっと残念。

◆12月号でペプシの“当たる”カンの話を載せてもらったのですが，そのハガキをポストに入れた数日後の秋も深まったころに，ペプシさんからTシャツが届きました。ペプシの方どうもすみませんでした。どうやら拾う神は2人もいたようです。　山下　寛(21)長崎県
Oh!Xのプレゼントのほうは届きましたか。感想を楽しみにしてますね。

◆1993年はいろいろとあったけど，なんといっても蚊に始まり蚊に終わった1年でした。1月から12月まで四季を問わずやられました。それにしてもなぜ12月まで蚊が飛んでいるのだろう。
藤原　彰人(23)岡山県
そういえば蚊の寿命ってどのくらいなんでしょう？

◆雨の降る夜，ガスのメーターが故障してガスが来なくなりました。次の日は晴れたので元どおりになりました。なぜでしょう（実話）。
武田　泰法(21)長崎県
ということは梅雨のときだと……。もしもお湯をガスで沸かすタイプだとお風呂に入れ……。

◆KOさんへ。私も以前似たような経験をしているので思わず大爆笑してしまいました。ウチもユニットバスで，通気が悪くよくカビが生えるのです。そこでカビキラーを壁という壁すべてにブチまけ，なんとそのままシャワーを浴びてしまったのです。換気扇があるから大丈夫だろうと思った私がバカでした。「ここで死ぬのか……」とマジで思いました。



◀木村　知史　神奈川県
とっても季節はずれだけど，去年は夏がほとんどなかったから，冬でも夏の気分をちょっと味わいたいなと思って載せてみました。

堂領　輝昌(19)神奈川県
みんな似たようなことをしてるんですね。

◆最近，あの名曲「きんたの大冒険」がリメイクして登場するらしい。近藤　真司(22)新潟県
それはぜひ聴いてみないと……。

◆いまは冬だけど，僕の気持ちはすっかり春！でも，すぐに秋が来て，冬になってしまうかも……。　小山　優一(20)東京都
えー，夏はないんですか。冬はなくてもいいから夏が……。それにしても寒い。

ぼくらの掲示板

●掲載ご希望の方は，官製ハガキに項目(売る・買う・氏名・年齢・連絡方法……)を明記してお申し込みください。
●ソフトの売買，交換については，いっさい掲載できません。
●取り引きについては当編集部では責任を負いかねます。
●応募者多数の場合，掲載できない場合もあります。
●紹介を希望されるサークルは必ず会誌の見本を送ってください。

売ります

★24ドット熱転写カラー漢字プリンタ「CZ-8PC3-BK」を5,000円で売ります。送料込み，ただしケーブルなし，1ドット印字ヌケ（字が読めないということはない)。連絡は往復ハガキでお願いします。〒867　熊本県水俣市袋8-23-3　高平雅由(34)

★48ドット熱転写カラー漢字プリンタ「CZ-8PC5」を送料込み30,000～35,000円で売ります。箱，マニュアル，付属品などすべてあります。色は黒です。連絡は希望価格を明記のうえ，官製ハガキでお願いいたします。〒457　愛知県名古屋市南区中割町4-89　県営中割住宅404号　神野力(18)

★アイテックのX68000用SCSIハードディスク「TX-180（180Mバイト）」を30,000～40,000円程度で売ります。箱，説明書なし，ケーブルあり。3年ほど使用しました。動作中の音がうるさいですが完動です。連絡は希望価格を明記のうえ，往復ハガキでお願いします。〒769-01　香川県綾歌郡国分寺町新名176-6　浅野　公昭(22)

★サイバースティック「CZ-8NJ2」を10,000円（送料別）で売ります。箱，説明書，付属品あります。1年使用，実質使用は3カ月です。連絡は往復ハガキでお願いします。〒343埼玉県越谷市南荻島3321-1　荒川　享(20)

★アクセスのX68000用MS-DOSエミュレータ「CONCERTO-X68K」を25,000円で売ります。連絡は官製ハガキでお願いします。〒167　東京都杉並区上井草2-26-9　秋山　和徳(22)

買います

★X68000用SCSIボード「CZ-6BS1」を13,000円（送料込み）で譲ってください。連絡は往復ハガキでお願いします。〒193　東京都八王子市横川町636-9　藤沢　実(20)

★アイ・オー・データ機器のX68000用増設RAMボード「PIO-6BE2-2ME」を18,000円，もしくは「PIO-6BE4-4ME」を32,000円前後で買います。送料は当方が負担します。連絡は往復ハガキでお願いします。〒674　兵庫県明石市大久保町松蔭374-5　敏森　健裕(21)

★X68000用1Mバイト増設RAMボード「CZ-6BE1」を送料込み12,000～17,000円程度で買います。完動品で，箱，説明書，付属品があるものを希望します。連絡は往復ハガキでお願いします。〒355-01　埼玉県比企郡吉見町上銀谷81-2　井上　政広(24)

★RolandのGSサウンドモジュール「SC-55mkII」を40,000円くらいで譲ってください。箱はなくても構いませんが付属品，説明書つきの完動品に限ります。なるべく安く譲ってくれる方を優先します。連絡は往復ハガキでお願いします。〒355-01　埼玉県比企郡吉見町上砂528-1　河上博仁(17)

バックナンバー

★Oh!X1990年7月号を送料込み2,500円で買います。広告の切り抜きは問題ありません。連絡は官製，往復ハガキどちらでも結構です。〒321-12　栃木県今市市今市246-6　福田　安章(16)

編集室から from E・D・I・T・O・R

DRIVE ON

このコーナーでは，本誌年間モニタの方々のご意見を紹介しています。今月は12月号の内容に関するレポートです。

●進化しているのはハードだけではないのだ，そう感じます。ソフトウェア技術とデザイン。特にデザインは，多色に慣れたデザイナーが増えたためか格段によくなったと思います。ソフト技術も，ありがちな演出が多いなか「エトワール・プリンセス」や「悪魔城ドラキュラ」などが新しい可能性を提示しました。どちらのソフトもその世界観に合った演出が多くのプレイヤーを魅了したことでしょう。

衝撃の「リッジレーサー」が出現しようとも，目の前にあるハードでアルゴリズムを駆使し，コードを極めた人たちが築き上げた文化。それがX68000のゲームなのでしょう。

コンシューマーでは，「スターフォックス」や「シルフィード」などが，家庭用では夢とされてきたサーフェイス表現を実現しました。ゲームデザインなどの面でまだまだ課題はあるようですが，それでも可能性を見せてくれたこの2作品。特に「シルフィード」は専用チップではなくアルゴリズムの勝利。どんなハードが出現しようとも最後に頼れるのは「アルゴリズムの可能性」なのでしょうか。

さあ，あとはゲームデザイン。こればかりは感性の問題だと思います。「アルゴリズムの可能性」をいかにうまく使うか。ゲームデザインには進化ではなく「突然変異」を望んでいるのは私だけでしょうか。

中矢　史朗(23)　X68000ACE-HD　愛媛県

●マーケティングからは，まったく新しい魅力ある商品（ゲーム）が創り出されることはないでしょう。先頭を走り続けること，オリジナリティをうち出していくことは本当に大変です。しかし，X68000は多くのプログラミングユーザーを擁しています。この素晴らしいハードを舞台に活躍するゲームクリエイターが現れる可能性は高いと思います。ライターの皆さんの“熱い”記事を読んでそう感じました。

橋本　和典(26)　X68000XVI　東京都

●「リッジレーサー」。ここまでくると，とてもゲームとは思えないくらいです。実際にプレイしてみましたが，技術のすごさに目を奪われてゲームどころではありませんでした。最近はゲームが面白いかどうかよりも，これはどうやって作ってあるのかということが気になり始めたので，今回の特集は参考になりました。

森崎　剛(21)　X68000XVI，PC-9801RX21　広島県

●最近になってX68000のソフトが活気をとりもどしてきたのがうれしい。本数は多くないのだが，えらく質が高い。どれを選んでもハズレはなしといいきっても過言ではない。

そんななかで，いまはやりの格闘ゲーム「ストリートファイターIIダッシュ」のできは「さすがカプコン！」といったものであった(CPSファイター使用)。「餓狼伝説2」は，これを書いている時点でまだ発売されていない。これは移植度も気になるが，注目すべきは価格設定である。初回のみではあるものの，パッドがついて9,800円はなかなかのサービスぶりである。正規ユーザーへのこのような差別化は実によいことだと思う。今後，ほかのメーカーにもまねをしていただきたい。

吉岡　洋明(20)　X68000 PROII，PC-8801MA，FM-NEW7　埼玉県

●CDの最大録音時間と同じだけの映像と音を記録できる！「VIDEO CD」はすごい。DCTでは画質は悪いけど，動画ならたいして気にならない。うーん，高くなければいいけど。

内藤　陽一(26)　X68000，PC-9801NS/E　東京都

●「MATIER」がさらに強力になったそうで。使ってみたいけれどもX68000を持っていないので使えない。ところで，以前から不思議に思っていたけれど「MATIER」などでX68000はプロにも使われているのに，どうして24ビットフルカラーのフレームバッファが発売されないのだろう。某国民機にすらあるのに……。その点，AMIGAはたくさんの種類のフレームバッファが発売されています。オパールビジョンはよいですよ。

松永　孝治(23)　X1turbo model30，PC-9801N，AMIGA1200/85MB　鳥取県

●「MATIER」の紹介記事の最後のほうで，「CG描くならメモリは湯水と思え」には参りました。でも，Macintoshのグラフィックソフトはメモリが10Mバイト以上ないと使えないよ！なんていってるのですから，Macintoshのソフトがウィンドウ上で動かすことを割り引いても，たった4Mバイトでこれだけの機能が動作するというのは立派だと思います。

野原　賢次(32)　X68000 ACE-HD，X1turbo model30　埼玉県

●DōGA CGアニメーション講座はいつものことながら説明が丁寧で好感がもてます。X68000の画面モードに関することなど，知ってて当たり前のようなことまでしっかりフォローされています。それにパソコン操作に慣れていれば，見ただけでわかるような操作まで細かく説明してあります。素晴らしいのひと言です（……というのは少々大げさかな)。こんな感じのプログラミングの入門講座みたいな連載を始めてもらえると，プログラミングを楽しめる人がもっと増えると思うのですが……。

林　大助(18)　X68000SUPER，PC-8801mkII FR　神奈川県

●「X-OVER・NIGHT」は好きな連載だったので最終回なのはとても残念です。コンピュータだけにとらわれない話題の豊富さは，Oh!Xのなかでは貴重な存在だったと思います。いずれ別の連載で会えることを楽しみにしています。長い間，ご苦労さまでした。

北風　保(22)　X68000 ACE　東京都

ごめんなさいのコーナー

12月号　THE SOFTOUCH

P.26　同ページに掲載されたカプコンの電話番号は，同社の新製品情報の電話番号でした。そのため，X68000版の「ストリートファイターIIダッシュ」に関しての問い合わせを行うことはできません。ご迷惑をおかけしましたことをお詫びいたします。

バグに関するお問い合わせは
☎03(5642)8182(直通)
月〜金曜日16：00〜18：00

お問い合わせは原則として，本誌のバグ情報のみに限らせていただきます。入力法，操作法などはマニュアルをよくお読みください。

また，よくアドベンチャーゲームの解答を求めるお電話をいただきますが，本誌ではいっさいお答えできません。ご了承ください。

習うより慣れろ！とりあえず実行

▼X680x0シリーズに標準添付されている開発言語，それがX-BASICです。今月は，X680x0を所有していれば，誰にでも触れることのできるX-BASICを使ってグラフィックをいろいろと操作してみました。

特集記事で掲載したリスト自体は，どれも短いのでそれほど苦労することもなく入力できると思います。なんとなくでも興味をもったものがあれば，とりあえず打ち込んで実行させてみてください。

もしも，わからないところや疑問点などがあれば，自分の手でマニュアルと格闘してみましょう。あくまで，自分の手でどこが問題なのか，なにが知りたいのかを明確にすることが必要なのです。完全にとはいかなくても，なんとなく問題を解決できれば，その悩んだ分が経験として必ず身につきます。

もしも解決できない場合でも，同じような疑問は別のときにまた現れるでしょう。そのときには，漠然と理解していたものがより明確になるはずです。思っていたことが間違っていたとしても，その都度訂正すればいいのですから，ものおじしていてはいけません。

やってみたいな，と思ったらすぐに行動してみましょう。そして，ぜひプログラミングという世界に足を踏み入れてみませんか。

▼さて，毎年恒例のGAME OF THE YEARの投票があります。1993年のラストには，ビックタイトルが立て続けにリリースされました。それだけでなく1993年前半に発売されたソフトも忘れないでくださいね。それから表にないノミネート作品以外のゲームについてや，ソフトハウスへのメッセージなどでも結構です。とにかく，ここでゲームを語らずしていつ語るのですか。皆さんの熱いメッセージをお待ちしています。

応募方法は昨年と同じで，アンケートハガキを使用する場合には，各部門ごとに作品名を挙げてください。締め切りは2月18日必着ですから，忘れずに送ってください。いまから4月号の発表が楽しみですね。

投稿応募要領

- 原稿には，住所・氏名・年齢・職業・連絡先電話番号・機種・使用言語・必要な周辺機器・マイコン歴を明記してください。
- プログラムを投稿される方は，詳しい内容の説明，利用法，できればフローチャート，変数表，メモリマップ（マシン語の場合）に，参考文献を明記し，プログラムをセーブしたテープ（ディスケット）を添えてお送りください。また，掲載にあたっては，編集上の都合により加筆修正させていただくことがありますのでご了承ください。
- ハードの製作などを投稿される方は，詳しい内容の説明のほかに回路図，部品表，できれば実体配線図も添えてください。編集室で検討のうえ，製作したハードが必要な場合はご連絡いたします。
- 投稿者のモラルとして，他誌との二重投稿，他機種用プログラムを単に移植したものは固くお断りいたします。

あて先

〒103 東京都中央区日本橋浜町3-42-3

ソフトバンク出版部

Oh!X「㊢㊀㊦㊔」係

SHIFT・BREAK

▶ボーナスでコンタクトレンズを買った。まだ1週間くらいだから，装着していると目のなかでレンズが動いているのがわかる。しかし，すでに2回失くしそうになった。2日目に洗面所で下水道に流しそうになり，4日目に電車の中で床に落とした。空いてたからすぐ見つかったけど，満員電車で誰かに踏まれたら……泣くに泣けないよねぇ。 (Y.A.)

▶某月某日。3日ぶりにアパートへ戻ると2階からの水漏れで部屋の一部が水浸し。分解したまま放ってあったXVIのことを思い出して一瞬ヒヤッとしたが，幸い布団と電化製品は無事であった。ただOh!Xのバックナンバーを半壊状態にされたのは許し難いものが……。場所的には気に入っているので引っ越したくはないが，そろそろ潮時か。 (進)

▶腕時計が死んだ。止まる前の3カ月ほどは2，3時間の遅れは当たり前，いつの間にか止まってることも。いつも時間を直してやらないとならなかったし，終電を逃し，野宿も1度や2度じゃなかった。でも，そんなやっかいなところもあの時計なら許せた。ところで原稿書き用のPC-9801の調子が悪い。いっとくけど，君の場合は許されないからね。(で)

▶夢を見た。ジェームス・ブラウンと一緒にオリジナル・ラヴのセッションをやっていた（わかる人にはわかるこのオカシサ）。アセりまくってやったこともないドラムを必死に叩いていると，J.B.から「モア，ファンク!!」の声。とたんにみなぎるパワー，流れ出すアツい16ビート。も，もしかして，コレは何かの啓示か？ (E.K.)

▶AMIGAにアクセラレータをつけた。小さなスロットに50MHzの030と882とSIMMとSCSIを収める実装技術に感動。レンダリングも速い。が，よく熱暴走するので冷却ファンを買ってきて貼りつけたら本体からはみ出した。みっともない。こりゃ設計ミスだよ。おまけにWORLD CIRCUITが動かなくなった。がっかり。まあ70点くらいの買い物だったな。(A.T.)

▶名古屋港にある水族館に行って来た。さまざまな水族館のおいしいところを寄せ集めて作ったような。ウリはウミガメとペンギン。新しい施設はどこもヘンだ。昔ながらの設計と，ハイテクな施設が混沌として，ローテクで統一された場末の水族館より違和感を感じる。小手先のハイテクではなく，ディテールにまで凝ったコンセプトが必要なのだ。 (K)

▶編集長がしばらくサッカーの話を書かないというので書いてしまおう。実は，キャプテン翼の愛蔵版を毎月買っている。翼はふらのとの試合では残り5秒で得点して勝利した。連載時はちょっと大ゲサかと思ったが，去年のワールドカップ予選のあっけない幕切れを見せつけられたいまでは，素直に感動してしまう。やはりサッカーは面白いと思う。 (KO)

▶腕時計をするようになった。時計を持ち歩いていたのは受験生のときだけだろう。だって街に時計はあふれているし，周囲に時計がみあたらなければ人に聞けばいいしね。とりあえずデザインはシンプルで気に入っている。あ，でもJリーグの公式腕時計にすればよかったかな。そうすれば某編集長のようにJリーグのチケットがほかの人から……。 (高)

▶窓際の席からは高速道路を走る車が見える。その下を流れる川も，犬の散歩も見える。「散らかし魔はお客さんから隠す」という編集長の温かいご配慮で決められた席にはとても満足している。ああ，それなのに……。冬の夜長は暖房が切られ，ここは極寒の地。積み上げられた諸々がないともっと寒いに違いない。で，おかたづけはしないのさっ。 (ふ)

▶ある，ファン感謝デーでの1日。朝一でモーニングを取る。幸先のいいスタートだ。しかし，そのあと946プレイのハマリ（この時点で-28,000円）。やっぱり「SOLEX」怖いね。で，パチスロをあきらめ，パチンコのシマへ行く。2,000円でフィバー。これもラッキー。しかし追加投資25,000円。結局，1日の負け額を更新して家路についた（涙）。 (J)

▶寒い。先月末のこと，管理室に暖房を強くしてくれというと「じゃあ空調を切りましょう」「？」「まだシステムが暖房に切り換わってないんですよ」「じゃあ，いま動いてるのは冷房ですか」「そういうことになります」以来，一昨日は初雪も降ったというのに暖房が入った気配はまったくない。が，この事実を知る人は少なかった。 (なぜか窓際のU)

▶最近のWindows用日本語ワープロでは，1行の文字数を指定できるというのを売り文句にしているものが多い。ところがどの製品も1行の文字数を変えると，段落の幅は変わらずに文字の間隔が勝手に広がったり詰まったりする。これも欧米のアプリケーションの影響か。とするとEGWord SX-68Kはどうかなぁ。ちょっと心配。 (T)

microOdyssey

編集後記でも負け自慢っぽいことを書いていることでもわかるとおり，最近，僕の趣味となっているのがパチンコだ。巷に溢れかえっている雑誌を読みあさり，週末ともなればパチンコ屋に通う生活。デジタル回転数を数えたりなんかして，いっぱしのパチプロ気分に浸ることも多い。負けた話が多いが，たまには「CR花○開」で16連チャンという美味しい思いをしたこともある（トータルでは負けているが）。

まあ，ギャンブルだから勝つときもあれば負けるときも当然ある。しかし，パチンコの場合，対称となっているのは機械。いくら乱数で勝敗が決定されるとはいえ，その機械の中身を知ることで勝率を上げられるはず。なぜなら大当たり抽選確率，連チャン方法を把握することである程度の投資金額の基準，やめどきも機種に応じて決めることができるからだ。常に確実な情報と自分自身の体験をもとにパチンコ台を攻めていく。面倒臭いと思われるかもしれないが，自分の理論どおりに事が運んだときにはものすごく気持ちいい。そして勝つことの快感を覚えてしまうともうやめられない。

で，まず必要となってくるのがパチンコ情報誌。台の情報は，巷に溢れかえっているどの雑誌にも載っているが，なるべく多角的にたくさんの情報が載っているものがいい。あとは，現実に打ってみてデジタル回転数をカウントして大当たりまでの回転数データを集めたりして，勝つためのデータ分析をしてみるのだ。そして，自分で正しいと思ったことをとりあえず実践するのだ。それによって勝率が上がればもうけもの。それにごくたまに強力な破壊力をもつ攻略法が発覚することがある。確かに雑誌の情報を過信するのは禁物だが，美味しい情報を見逃すことを考えれば，雑誌代くらいけちらないほうがいい。

といっても，雑誌で紹介されている攻略記事は，店側の対策がすでに行われていたり，ある特定の裏基板（または裏ROM）のみのものが多い。そのため，攻略法を使った美味しい記事を読むたびに台を探すが見つからず，悔しい思いをしてきたのも事実だ。

しかし，実際にその攻略法を使える機会がやってきたのだ！　最初は半信半疑で実行していたが，それが通用するとわかったときの快感は，いままでギャンブルで経験したどれよりもすごかった。なにしろ，店員にさえ気をつけていれば半永久的に大当たりを引き続けることができるので，その威力は想像を絶する。攻略法を1日行えば，10万円どころか30万円ペースで稼ぎ出せるというふれこみどおり，まさに出まくりなのだ。

最終的に僕自身が体験した記録は，最大16連チャン，1日で28箱（結局疲れたのでやめた），金額にして14万円也。はっきりいって真面目に働くのが馬鹿らしくなるくらいの金額を，たった1日で手にすることができたのだから。もしも，これからパチンコを始めようなど思っている人は，必ず，打とうとするパチンコ台の情報を事前にチェックしておこう。もしかしたら，儲けることができるかもしれない。

結局，対策が行われるまでの5日間で3カ月分の家賃を叩き出し，充実した正月を過ごせた。本当にごちそうさまでした。（J）

1994年3月号2月18日(金)発売

ひなまつりPRO-68K

- ポリゴナイザライブラリ，SLASH ver.2.0
- グラフィックツールの拡張，Z's-EX，MAT-EX
- SCSIドライバ用アニメーションツール
- そのほかゲームやツールいろいろなデータ満載

新製品紹介

H.A.R.P for MC68000/Video PC for X680x0

特別付録　5"2HDディスク　　予価800円

バックナンバー常備店

東京	神保町	三省堂神田本店5F	03(3233)3312
	〃	書泉ブックマートB1	03(3294)0011
	〃	書泉グランデ5F	03(3295)0011
	秋葉原	T-ZONE 7Fブックゾーン	03(3257)2660
	八重洲	八重洲ブックセンター3F	03(3281)1811
	新宿	紀伊国屋書店本店	03(3354)0131
	高田馬場	未来堂書店	03(3209)0656
	渋谷	大盛堂書店	03(3463)0511
	池袋	旭屋書店池袋店	03(3986)0311
	八王子	くまざわ書店八王子本店	0426(25)1201
神奈川	厚木	有隣堂厚木店	0462(23)4111
	平塚	文教堂四の宮店	0463(54)2880
千葉	柏	新星堂カルチェ5	0471(64)8551
	船橋	リブロ船橋店	0474(25)0111
	〃	芳林堂書店津田沼店	0474(78)3737
	千葉	多田屋千葉セントラルプラザ店	0472(24)1333
埼玉	川越	黒田書店	0492(25)3138
	川口	岩渕書店	0482(52)2190
茨城	水戸	川又書店駅前店	0292(31)0102
大阪	北区	旭屋書店本店	06(313)1191
	都島区	駸々堂京橋店	06(353)2413
京都	中京区	オーム社書店	075(221)0280
愛知	名古屋	三省堂名古屋店	052(562)0077
	〃	パソコンΣ上前津店	052(251)8334
	刈谷	三洋堂書店刈谷店	0566(24)1134
長野	飯田	平安堂飯田店	0265(24)4545
北海道	室蘭	室蘭工業大学生協	0143(44)6060

定期購読のお知らせ

Oh!Xの定期購読をご希望の方は綴じ込みの振替用紙の「申込書」欄にある『新規』『継続』のいずれかに○をつけ，必要事項を明記のうえ，郵便局で購読料をお振り込みください。その際渡される半券は領収書になっていますので，大切に保管してください。なお，すでに定期購読をご利用の方には期限終了の少し前にご通知いたします。継続希望の方は，上記と同じ要領でお申し込みください。

基本的に，定期購読に関することは販売局で一括して行っています。住所変更など問題が生じた場合は，Oh!X編集部ではなくソフトバンク販売局へお問い合わせください。

海外送付ご希望の方へ

本誌の海外発送代理店，日本IPS(株)にお申し込みください。なお，購読料金は郵送方法，地域によって異なりますので，下記宛必ずお問い合わせください。

日本IPS株式会社
〒101 東京都千代田区飯田橋3-11-6
☎03(3238)0700

Oh!X　2月号
■1994年2月1日発行　定価600円(本体583円)
■発行人　橋本五郎
■編集人　稲葉俊夫
■発売元　ソフトバンク株式会社
■出版事業部　〒103 東京都中央区日本橋浜町3-42-3
Oh!X編集部　☎03(5642)8122
販売局　☎03(5642)8100 FAX 03(5641)3424
広告局　☎03(5642)8111
■印　刷　凸版印刷株式会社

バックナンバー案内

BACK ISSUES

ここには1993年2月号から1994年1月号までをご紹介しました。現在1992年6, 7, 9, 12, 1993年6～12, 1994年1月号の在庫がございます。バックナンバーはお近くの書店にご注文ください。定期購読の申し込み方法は148ページを参照してください。

1993



2月号（品切れ）
特集　画像創造のために
連載　DōGA CGアニメーション講座/マシン語プログラミング
響子 in CGわ～るど/ショートプロ/よいこのSX-WINDOW
ハードエ作/吾輩はX68000である/Computer Music入門
●新製品紹介　Communication SX-68K
LIVE in '93　FIRE CRACKER/サンバDEグワッシャ!
THE SOFTOUCH　極/ドラゴンスレイヤー英雄伝説/
機甲装神ヴァルカイザー/キングス・ダンジョン
全機種共通システム　BLACK JACK



3月号（品切れ）
特集　X-BASICを学ぶ
連載　DōGA CGアニメーション講座/マシン語プログラミング
響子 in CGわ～るど/ANOTHER CG WORLD/ハードエ作
ショートプロ/Computer Music入門/Z80's Bar
●緊急速報　32ビットマシンX68030
●新製品紹介　音源モジュールSC-33/GS音源搭載JW-50
LIVE in '93　ストリートファイターII/晴れたらいいね 他
THE SOFTOUCH　究極タイガー/チェルノブ/シムアント 他
全機種共通システム　シューティングゲームコアシステム作成法(1)



4月号（品切れ）
特集　X68第7世代へ
連載　DōGA CGアニメーション講座/マシン語プログラミング
響子 in CGわ～るど/ショートプロ/よいこのSX-WINDOW
ハードエ作/吾輩はX68000である/Computer Music入門
●決定! 1992年GAME OF THE YEAR
●名作ゲーム再遊記
LIVE in '93　FIGHTMAN/ミンキーモモより 愛しのマーシカ
THE SOFTOUCH　スターフォース/元朝秘史 他
全機種共通システム　シューティングゲームコアシステム作成法(2)



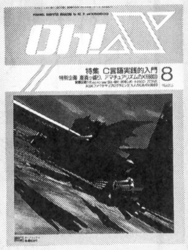
5月号（品切れ）
特集　襲撃! SX-WINDOW
第8回　言わせてくれなくちゃだワ
連載　DōGA CGアニメーション講座/ANOTHER CG WORLD
響子 in CGわ～るど/ショートプロ/大人のためのX68000
ハードエ作/吾輩はX68000である/Computer Music入門
●X68030へのソフトウェア対応について
LIVE in '93　MAGICAL SOUND SHOWER/もう笑うしかない 他
THE SOFTOUCH　エトワールプリンセス/メガロマニア 他
全機種共通システム　シューティングゲームコアシステム作成法(3)



6月号
創刊11周年特別企画　確率遊技シミュレーション
連載　DōGA CGアニメーション講座/こちらシステムX探偵事務所
響子 in CGわ～るど/ショートプロ/大人のためのX68000
ハードエ作/吾輩はX68000である/Computer Music入門
●新製品紹介　SC-55mkII
LIVE in '93　ストリートファイターIIより 春麗のテーマ/
BAY YARD/LOVE&CHAIN
THE SOFTOUCH　餓狼伝説/信長の野望・覇王伝 他
全機種共通システム　REVERSI



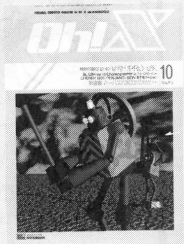
7月号
特集　席巻するローテク文明
連載　DōGA CGアニメーション講座/こちらシステムX探偵事務所
響子 in CGわ～るど/ショートプロ/マシン語プログラミング
ハードエ作/吾輩はX68000である/Computer Music入門
●新製品紹介　ドローイングパット33070&MATIER
LIVE in '93　Midnight Circle/今日の日はさようなら/赤い靴
THE SOFTOUCH　悪魔城ドラキュラ/リブルラブル/大航海時代II/
銀河英雄伝説III/幻影都市/ヴェルスナーグ戦乱
全機種共通システム　MSX用S-OS "SWORD"



8月号
特集　C言語実践的入門
連載　DōGA CGアニメーション講座/こちらシステムX探偵事務所
響子 in CGわ～るど/Computer Music入門/大人のためのX68000
吾輩はX68000である/ショートプロ/ANOTHER CG WORLD
●特別企画　夏真っ盛り, アマチュアリズムのX68000
LIVE in '93　SPLASH WAVE
THE SOFTOUCH　悪魔城ドラキュラ/リブルラブル/餓狼伝説/
ロボットコンストラクションR.C./Winning Post
全機種共通システム　MACINTO-C再掲載



9月号
特集　光学式磁気円盤MO
連載　DōGA CGアニメーション講座/こちらシステムX探偵事務所
響子 in CGわ～るど/ショートプロ/大人のためのX68000
ハードエ作/Computer Music入門/ANOTHER CG WORLD
●新製品紹介　OS-9/X68030
LIVE in '93　ファイナルファンタジーVのテーマ/銀河鉄道999/
アルスラーン戦記IIより 汗血公路/ちょうちょ
THE SOFTOUCH　悪魔城ドラキュラ/コットン/ダーク・オデッセイ 他
全機種共通システム　7並べ/SLANG再々掲載



10月号
特別企画　秋祭りPRO-68K
連載　ハードコア3D/Computer Music入門/マシン語プログラミング
DōGA CGアニメーション講座/こちらシステムX探偵事務所
響子 in CGわ～るど/ショートプロ/吾輩はX68000である
●特別付録　秋祭りPRO-68K（5"2HD）
●SCSIパックンTOWER JACK
LIVE in '93　未来予想図II/OutRunより PASSING BREEZE
THE SOFTOUCH　コットン/The World of X68000/あにまーじゃんV3
全機種共通システム　シューティングゲームコアシステム作成法(4)



11月号
特集　ポリゴナイザSLASHの活用
連載　ハードコア3D/Computer Music入門/ファイル共有の実験と実践
こちらシステムX探偵事務所/目指せジョイスティックの星
響子 in CGわ～るど/ショートプロ/大人のためのX68000
●新製品紹介　Easydraw SX-68K
OS-9 Ultra C/Technical Tool Kit
LIVE in '93　渚のアデリーヌ/エロティカ・セブン
THE SOFTOUCH　ぶたさん/ダイアット・ヴァークス
全機種共通システム　S-OSで学ぶZ80マシン語講座(1)



12月号
特集　古今東西ゲーム議論
連載　ハードコア3D/マシン語プログラミング/響子 in CGわ～るど
DōGA CGアニメーション講座/こちらシステムX探偵事務所
ショートプロ/Computer Music入門/ファイル共有の実験と実践
●新製品紹介　MATIER ver.2.0
C Compiler PRO-68K ver.2.1 NEW KIT
LIVE in '93　クリスマス・イブ/星に願いを
THE SOFTOUCH　ネメシス'90改/項劉記/スーパーリアル麻雀PII&PIII
全機種共通システム　エディタアセンブラREDA再掲載

1994



1月号
特集　Z-MUSICシステムver.2.0
連載　ハードコア3D/ゲーム作りのKNOW HOW/響子 in CGわ～るど
DōGA CGアニメーション講座/こちらシステムX探偵事務所
ショートプロ/Computer Music入門/ファイル共有の実験と実践
●特別企画　ANOTHER CG WORLD in Hong Kong
LIVE in '94　LAST WAVE/スターウォーズ/明日への扉/夢路より 他
THE SOFTOUCH　ストリートファイターIIダッシュ/餓狼伝説2/
ドラゴンバスター/X68000傑作ゲーム選
全機種共通システム　S-OSで学ぶZ80マシン語講座(2)



満開の電子（でんこ）ちゃん

作・え 岡村祭



※Vol.75（94年8月号）以降は、毎月2枚組1,500円（本体1,456円）に価格改訂される可能性がございます。ご了承の上ご送金ください。

講読方法：定期購読もしくはソフトベンダーTAKERUでお買い求めいただけます。

★定期購読の場合＝購読料６ヶ月分6,000円（送料サービス、消費税込）を、現金書留または郵便振替で下記の宛先へお送り下さい。

現金書留の場合：〒171 東京都豊島区長崎1-28-23 Muse西池袋2F ㈱満開製作所

郵便振替の場合：東京 5－362847 ㈱満開製作所

- ご注文の際は、郵便番号・住所・氏名・電話番号を忘れずに記入して下さい。
- 3.5インチディスク版をご希望の方は、「3.5インチ版」とご指定下さい。
- 新規購読の方は「新規」と明記して下さい。なお、特に購読開始号のご指定がない場合は既刊の最新号からお送りいたします。
- 製品の性格上返品には応じられませんが、お申し出があれば定期購読を解約し残金をお返しします。

★TAKERUでお求めの場合＝１部につき1,200円（消費税込）です。

- 定期購読版と内容が一部異なる場合があります。御了承下さい。
- お問い合わせ先 TEL(03)3554－9282（月～金 午前11時～午後６時）

（なお、定期購読版のバックナンバーについては定期購読の方のみご注文を承ります）

ボンジュール・マドモアゼール。ア・モーレモーレ・ムーア。奥さん、たゆたう午後の日差しの中、如何お過ごしでしょうか？奥さん、ご主人の帰りを待ちながらの一時のアヴァンチュール、奥さん、電脳倶楽部を読みながら、生活の疲れを癒しつつ、想いは遠くシャンゼリゼの並木、甘美なジゴロ達の愛の囁きの中、いいですか？奥さん。電話切らないで下さいね、奥さん、CMです。

電子「マウス一つでラクラク操作」

八名「まずい～もう一杯！」

奥さん、みのです。そんな亭主別れちゃいなさい。セシボーン。

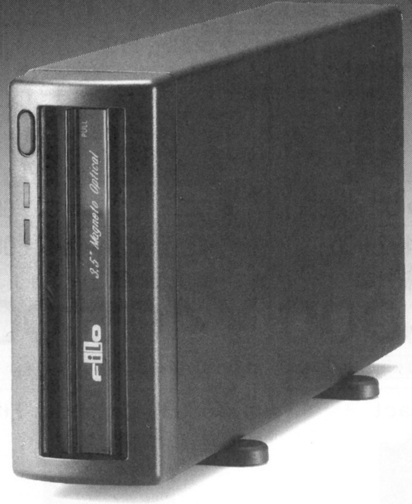


上村人妻殺し（広島県）

FiLo ファイロ

ブラックモデル現る!!

エアフィルタ交換不要の3.5インチ光磁気ディスクユニット



X680x0にジャストフィット

■世界最小クラスのコンパクトなボディ。縦置/横置可能。
■深夜でも気にならない低騒音ファンを使用。
■平均シークタイム30ms、回転数3600rpmの高性能ドライブ。

CS-M120PX　通販限定

本製品は、特別企画商品につき一般の小売店では購入できません。

標準価格 178,000円 → 特別価格 118,000円（税込）

●お申し込みはFAXまたは郵送にて

注文書の太枠線内にご記入の上郵送またはFAXにてお送りください。

お申し込み先
コパル綜合サービス株式会社　通販係
東京都板橋区志村2-16-20
TEL 03-3965-1144
FAX 03-3558-3229

●お支払いは銀行振込で

代金は下記の口座までお振込みください。
(振込手数料はお客様負担で電信扱でお振込ください。)

口座番号　東海銀行　板橋支店　当座預金160141
口座名義　コパル綜合サービス株式会社

・商品の引き渡しは代金お支払い後となります。
・商品はご入金確認後、原則として3日以内に発送致します。
(在庫切れの場合はご連絡いたします。)

◆今回お買い求めの方に限りケーブル*・ターミネータ・送料をサービス。
***ご注文の際にご希望のケーブルをご指定ください。**

◆SCSI I/Fボードはパソコン本体に付属のものまたは純正品が使用可能です。
その他サードパーティ製のSCSI I/Fボードとの接続についてはお問い合わせください。

X680x0以外のパソコン用接続キット、オプションも用意しております。

主な接続キット
●PC98接続キット
●Macintosh接続キット
●FM接続キット
●AT接続キット

※商品の技術的なご質問・ご相談は
ユーザーサポート係　TEL03-3965-1161

FiLo注文書　FAX 03-3558-3229

品名	CS-M120PX	ご注文台数　　台	ご連絡先 TEL（　） FAX（　）
ケーブル*1	□フル～ハーフ　□ハーフ～ハーフ		
お名前	フリガナ		
お届先住所	(〒　-　)　1．会社　2．自宅 都道府県　区市郡		

(弊社記入欄)
受付番号
受付日
納入日
備考

*1 どちらかご希望のケーブルをご指定ください。



マイコン専門ショップ

P&A

SHARPエキスパートショップ

今が購入のチャンス!

SHARP

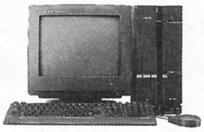
1/18〜2/17

注目!!平成6年4月末一括払い手数料(金利)無料(平成6年2月末/3月末/4月末のいずれかをご指定ください。)

X68030お買い得セット

(クレジット表:送料・消費税込み)

①X68030



●CZ-500C ●CZ-608D

定価合計¥492,800

P&A超特価

¥309,000

12回	28,200	24回	14,900	36回	10,400	48回	8,100	60回	6,800

②X68030 HD



●CZ-510C ●CZ-608D

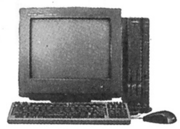
定価合計¥582,800

P&A超特価

¥409,000

12回	37,300	24回	19,700	36回	13,700	48回	10,700	60回	9,000

③X68030 Compact



●CZ-300C ●CZ-608D

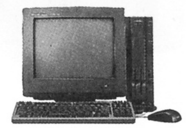
定価合計¥482,800

P&A超特価

¥340,000

12回	31,000	24回	16,400	36回	11,000	48回	8,900	60回	7,500

④X68030 Compact HD



●CZ-310C ●CZ-608D

定価合計¥572,800

P&A超特価

¥403,000

12回	36,700	24回	19,400	36回	13,500	48回	10,500	60回	8,900

■モニターの変更

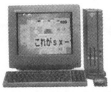
(※300シリーズにチューナー付のモニターを接続の場合CRTケーブルを購入して下さい。)

①CZ-607D(チューナー付)に変更の場合¥ 3,000
②CZ-614D(チューナー付)に変更の場合¥31,000
③CZ-621D(B)……………に変更の場合¥58,000
} 加算して下さい。

X68000 Compact XVI 旧シリーズ今が買いどき!!

(クレジット表:送料・消費税込み)送料¥2,000・消費税別

①本体+モニター



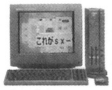
●CZ-674C-H
●CZ-608D-H

定価¥392,800

P&A超特価**¥162,000**

12回	14,800	24回	7,800	36回	5,400	48回	4,300	60回	3,600

②本体+モニター+FDD(5″×2)



●CZ-674C-H
●CZ-608D-H
●CZ-6FD5(FDD)

定価¥492,600

P&A超特価**¥209,000**

12回	19,100	24回	10,100	36回	7,000	48回	5,500	60回	4,600

③本体+モニター(TVチューナー付)



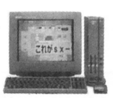
●CZ-674C-H
●CZ-614D-TN
●CZ-6CR1(RGBケーブル)
●CZ-6CT1(TVコントロール)

定価¥443,000

P&A超特価**¥199,000**

12回	18,200	24回	9,600	36回	6,700	48回	5,200	60回	4,400

④本体+モニター(TVチューナー付)+FDD(5″×2)



●CZ-674C-H
●CZ-614D-TN
●CZ-6CR1(RGBケーブル)
●CZ-6CT1(TVコントロール)
●CZ-6FD5(FDD)

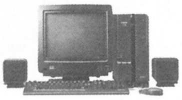
定価¥542,800

P&A超特価**¥247,000**

12回	22,500	24回	11,900	36回	8,300	48回	6,500	60回	5,400

X68000 XVI

①本体+モニター



●CZ-634C-TH(本体)
●CZ-608D-H(モニター)

定価¥462,800

P&A超特価**¥213,000**

12回	19,500	24回	10,300	36回	7,100	48回	5,600	60回	4,700

モニター変更の場合

※Compact XVI ①・②/XVI ①のモニターを、

● CZ-607D-TN(定価¥99,800)に変更の場合¥3,000加算して下さい。

● CZ-621D(B)(定価¥168,000)に変更の場合¥58,000加算して下さい。

X68030 発売記念

X68030をモニターとセットで/単品で } 購入の方

さらに現在お持ちのパソコンと、下取り交換されたお客様に期間中もれなく、

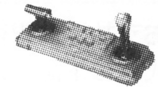
①サイバーステック (CZ-8NJ2 ¥23,800)

②X-68000 フロッピーアタッシュケース (¥8,000)

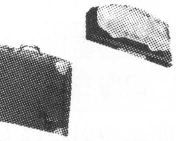
とクリスタルポルシェ (¥8,000)

以上のいずれかプレゼント!!

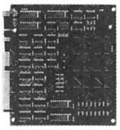
①



②



X68030/68000メモリボード(I/Oデータ)



①SH-5BE4-8M(X68030用)……………(送料・消費税込み¥47,586)特価¥45,500

②SH-6BE1-1ME(600C専用)…………(送料・消費税込み¥12,669)特価¥11,600

③1MB増設RAMボード(ACE/PRO/PROII用)(送料・消費税込み¥12,669)特価¥11,600

④2MB増設RAMボード(拡張スロット用)・(送料・消費税込み¥24,411)特価¥23,000

⑤4MB増設RAMボード(拡張スロット用)・(送料・消費税込み¥40,170)特価¥38,300

モデム

(送料¥1,000)

マイクロコア ●MC-14400FX……………(定価¥46,800)▶特価¥34,500

富士通 ●FMMD-3111G…………(定価¥35,800)▶特価¥24,800

オムロン ●MD-24XT10V…………(定価¥29,800)▶特価¥22,500

●MD-96XT10V…………(定価¥46,800)▶特価¥32,000

アイワ ●PV-AF144V5…………(定価¥64,800)▶特価¥49,000

X68000/68030専用ハードディスク

(送料¥1,000・消費税別)

外付



■富士通

⊙FMHD-1201G(120MB、17ms)‥定価¥ 70,000▶特価¥49,800

⊙HD-K240(モッキンバード)(240MB、15ms)・定価¥ 79,800▶特価¥49,800

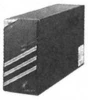
⊙HD-K540(モッキンバード)(540MB、10ms)・定価¥148,000▶特価¥98,000



■ロジテック

⊙SHD-FMX120(120MB)(ケーブル付)
……………………定価¥ 59,800▶特価¥47,000

⊙SHD-FMX240(240MB)(ケーブル付)
……………………定価¥138,000▶特価¥57,800



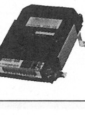
■ジェフ

⊙GF-240e(240MB/15ms/64K)・定価¥118,000▶特価¥ 49,800

⊙GF-340i(340MB/14ms/64K)‥定価¥158,000▶特価¥ 59,800

⊙GF-540i(540MB/8.5ms/256K)定価¥238,000▶特価¥108,000

内蔵



■CZ-500C/300C専用

⊙CZ-5H08(80MB/23ms)
……………………定価¥ 98,000▶特価¥71,800

⊙CZ-5H16(160MB/18ms)
……………………定価¥135,000▶特価¥99,500

●本広告の掲載の商品の価格については、消費税は含まれておりません。

安いのに親切
TSUKUMO

ツクモ決算セール!! 1/31まで

※棚ズレ、在庫限定品につき大特価で販売中!!
今すぐご来店下さい(品切れの際はご容赦下さい)

ツクモグローバルカード

好評
入会者受付中!



18才以上なら
学生でもOK!

〜国内・海外でも使える多機能カード〜

ジャックス・VISA、セントラル・マスターのカードです。
分割払い、ボーナス払いもOK!クレジット申し込みと同時にカード申し込みOK!お申し込みはTEL03（3251）9898又は各店で。
★各店頭では、JCB、日本信販、DC他各種カードも取り扱っております。

SHARP X68000/030シリーズ本体

お勧めの組み合わせ

CZ-500C-B ￥398,000
240MBハードディスク サービス
ツクモ特価￥340,000

CZ-300C-B ￥388,000
TS-XFDCA ￥ 9,800
ツクモ特価￥295,000



大好評
発売中!

SHARPは4Fに!

※満開製作所の商品も
取り扱っております!

X68000Compact XVI 24MHz改
RED ZONE ￥160,000
RED ZONE(2DD) ￥165,000
RED ZONE+MK-FD1 ￥180,000
満開製5インチFDD
MK-FD1 ￥39,800
MK-FD1カラーリングモデル ￥44,800

目玉商品!!

●年始特別御奉仕のX68000 台数限定●

X68000Compact XVI（CZ-674C）ツクモ特価￥99,800 66%OFF

NEW ツクモオリジナル TS-3XRシリーズ
〜X68000用外付ドライブ〜
3.5インチ
●2DD/2HD/2HC/1.44MBフォーマット対応
●CompactXVI/68030用ケーブル付
※Human68k Ver3.0以上が必要です。
※従来機種（フルピッチコネクタ）でお使いの方は別売ケーブル（TS-XR3CA特価￥3,500）が必要です。



TS-3XR1B 1ドライブ 定価￥33,800 **ツクモ特価￥26,800**
TS-3XR2B 2ドライブ 定価￥46,800 **ツクモ特価￥36,800**

限定販売中（無くなり次第終了となります。）
カラーイメージユニット接続ボックス TS-VTBOX 定価￥19,800 **ツクモ特価￥17,800**
CompactXVI/68030シリーズにカラーイメージユニットを接続する為のアダプターです。

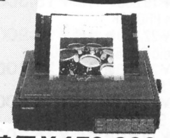
近日発売予定!
Music Card for X680x0(TS-6GM1)
予価 ￥39,800
MIDIインターフェースボードにGM規格の音源を搭載しました。
これ一枚で、MIDIコンピュータミュージックが楽しめます。

ビデオPC for X680x0 パッケージ
OS-9／X680x0上で作動する、MPEG動画再生の為のボード及びソフトのパッケージです。
マイクロウェア製
ツクモ特価￥57,000

●プリンター●

カラーイメージジェット
IO-735X-B（ケーブルセット） **ツクモ特価￥134,000**
バブルジェットプリンター
BJ-10VLite（ケーブルセット） **ツクモ特価￥ 39,800**

台数限定
★48ドットカラー熱転写プリンター
CZ-8PC5-BK
ツクモ特価￥39,800



カラーバブルジェットプリンター BJC-820J（ケーブルセット） 台数限定 **ツクモ特価￥170,000**

カラーイメージスキャナー
CZ-8NS1(限定品) **ツクモ特価￥ 79,800**
JX-220X **ツクモ特価￥135,000**
JX-325X **ツクモ特価￥152,000**

RGBシステムチューナー
CZ-6TU（-GY / -BK） **ツクモ特価￥23,800**

コンピュータアート

スーパーグラフィックツールセット
その1 慣れてしまうとマウスがいらない
DrawingPad........ ￥76,500
Matier Ver2.0........ ￥39,800
ツクモ特価￥95,000

スーパーグラッフィックツールセット
その2 ハイクオリティなのにこんなに安い
Desk Jet 505J Plus........ ￥78,000
プリンターケーブル........ ￥ 4,800
Matier Ver2.0........ ￥39,800
ツクモ特価￥89,000

通信販売のご注文は下記フリーダイヤルへ。
全国どこからでも通話料無料
受・注・専・用 フリーダイヤル **0120-377-999**
通販センター・・・03-3251-9911 商品についてのお問い合わせは各店または通販へ。

クレジット払い
月々￥3,000以上の均等払いも頭金なし。夏・冬ボーナス2回払いも受付中!

各種リース払い
くわしくは各店にお問い合わせ下さい。ケースに合わせてご相談承ります。

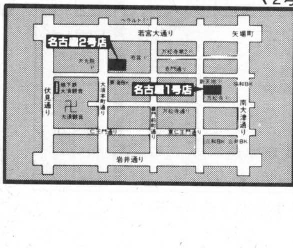
現金書留払い
〒101-91 東京都千代田区神田郵便局私書箱135号 ツクモ通販センター Oh!X係

カード払い（￥5,000以上）
通信販売でのご利用カード、ツクモグローバルカード、セントラル、ジャックス※ご本人様より電話で通販部へお申し込み下さい。

全国代金引換え配達
お申し込みはTEL03-3251-9911へお電話1本!配達日の指定もできます。

銀行振込払い
事前にTELでお届け先をご連絡下さい。
三和銀行 秋葉原支店（普）1009939
ツクモデンキ

ツクモIN名古屋
（1号店 第一アメ横ビル内 / 2号店 第二アメ横ビル内）





1号店 担当 横山



2号店 担当 松原

名古屋1号店 TEL052（263）1655 休 毎週火曜日
名古屋2号店 TEL052（251）3399 休 毎週水曜日

ツクモIN札幌
（ツクモ札幌店 / DEPOツクモ2番街店）





札幌店 担当 田口



DEPO店 担当 鈴木

札幌店 TEL011（241）2299 休 毎週木曜日
DEPO店 TEL011（242）3199 休 毎週木曜日

2・2倍速CD-ROMドライブ KGU-XCD2同等品 限定特価¥68,000で好評発売中!!（ドライバ別売り）

for X680x0 Series Only
オリジナル アプリケーション
開発速報#6

R&D Division
of
計測技研
FirstClassTechnology

1/15発売

SX-WINDOW用スケジュール管理ソフト
DoubleBookin'
標準価格 ¥12,800

SX-WINDOW だから、いっしょに暮らせる

SX-WINDOW。そのマルチタスク環境では、ユーザーの作業環境は一変します。より自由に、感性のおもむくままに。SX-WINDOWとDoubleBookin'の組み合わせで、あなたの暮しが、仕事が、変わります。

DoubleBookin'は、あたらしい考え方にもとづいたスケジュール管理ソフトです。

●多角的な予定設定/表示

予定はカレンダー、デイリー、グラフの3つのウィンドウで確認することができ、多角的に検討することができます。

何日にもまたがる長期の予定にも対応。短期の予定とあわせて、日常生活に即したスケジュール設定が可能です。

●マルチタスクをいかした豊富なイベント

予定を設定した時刻をメッセージやアラームで知らせるのはあたりまえ。DoubleBookin'は、SX-WINDOWのマルチタスク環境をいかして、様々なイベントを起こすことができます。音楽を演奏したり、テレビ画面に切り替えたり、シャーペンやEasy drawを起動したり…など、思うままのライフスタイル設計を可能にします。

●モジュール追加で成長し続けます

予定に設定したイベントの種類は、外部モジュールを追加することによってさらに増やすことができます。

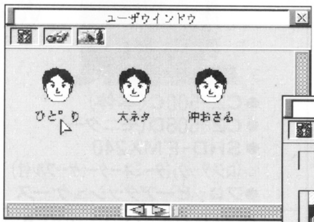
今後新しいアプリケーションや周辺機器が登場した場合でも、DoubleBookin'はそれらを取り込んで成長し続けます。

●電子手帳やシステム手帳とリンク

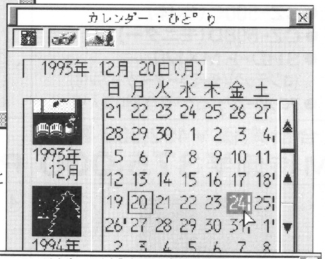
シャープ製電子手帳(DB-Z以上)とのスケジュールのやりとりを可能にしたほか、カレンダーやスケジュールのリフィル印刷もサポート。自宅やオフィスに縛られることなく、幅広いフィールドで活用できます。

●アンフィニーシステム社製MIC-68Kにも正式対応

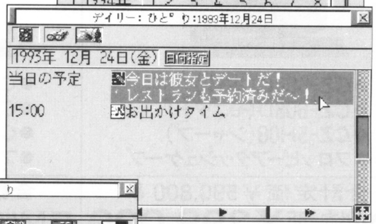
スケジュールに応じて、赤外線で家電製品をコントロールできます。



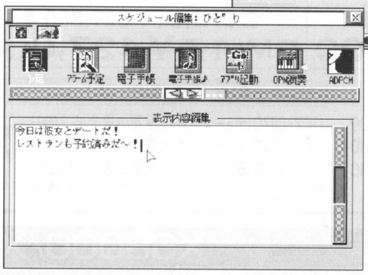
ユーザウィンドウ
ディスク容量が許す限り、何人のスケジュールでも登録可能。



カレンダーウィンドウ
スケジュールの設定状況が、ひと目で確認できるカレンダー表示。ここが作業の中心です。



デイリーウィンドウ
カレンダー上の日付をダブルクリックすると、一日のスケジュールを詳しく表示します。



編集ウィンドウ
スケジュールの入力はとてもかんたん。
イベントアイコンを選んで予定の内容を記入するだけ。

発売中
CD-ROM Driver Ver1.06
標準価格 ¥4,800

CD-ROM Driver Ver1.06は、Human68k上でCD-ROMをフロッピー感覚で扱えるようにするデバイスドライバです。一般的なISO9660フォーマットで記録されているCD-ROMなら、X680x0でそのまま読み出すことができます。

(1)ISO9660アクセス対応ドライブ
市販されているほとんどのCD-ROMドライブ

(2)CD-ROM XA対応ドライブ(SX-PhotoGallery対応)

東芝	XM-3301、XM-3401、およびその互換製品
ロジテック	LCD-500
松下電器	LK-RC533N25
弊社	KGU-XCD、KGU-XCD2

(3)オーディオ対応ドライブ

東芝	XM-3301、XM-3401、およびその互換製品
各社	SCSI-2コマンド対応製品
弊社	KGU-XCD、KGU-XCD2

発売中
SX-WINDOW用Photo-CDビュアー
SX-PhotoGallery
基本セット¥15,000

大切な写真をいつまでも美しく保存できるPhotoCD。SX-PhotoGalleryなら、PhotoCDのフルカラー記録をSX-WINDOW Ver.3.0のグラフィックウィンドウ上に再現できます。

SX-WINDOWの特長である、カット&ペーストによるアプリケーション間でのデータのやりとりにも対応。また、PhotoCDの画像展開モジュールはIVM.X用のリソースとして用意しましたので、キャンバス、シャーペン、Easydraw、EasypaintなどでPhotoCD画像を利用することができます。様々なソフトと連係し、力をあわせ、その結果のクオリティを追求するSX-WINDOWの思想を踏襲しています。

SX-PhotoGalleryの機能をすぐにお試しいただけるよう、Photo-CDのサンプル画像を収めた「Kodakフォトサンプラー」バンドルセット(定価¥19,000)もご用意いたしました。

※　SX-PhotoGalleryにはCD-ROM Driverが付属します。

発売中
X680x0用フリーソフトウェア集CD-ROM
FreeSoftwareSelection Vol.1
標準価格 定価 ¥5,000

※　記載されている会社名および商品名は各社の登録商標もしくは商標です。

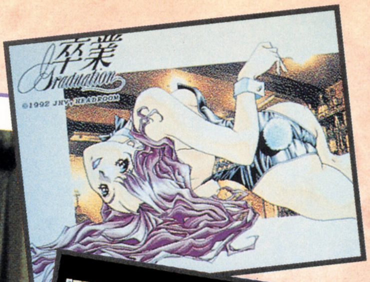
お求めはお近くのパソコンショップ、または弊社通販部(TEL：0286-22-9811)へお申し込みください。
通販ご希望の方は、ソフト代金＋送料¥1,000に消費税を加え、ご住所・お名前・電話番号・商品名を明記した紙を同封の上、現金封筒でお申し込みください。

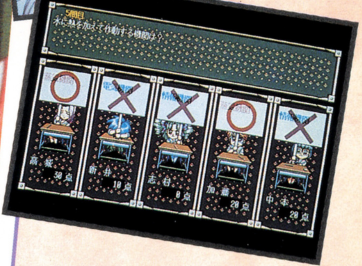
低金利クレジット　通信販売送料 全国一律¥1,000 長期クレジット可能
※表示価格に消費税は含まれておりません
株式会社 計測技研 マイコンショップBASIC HOUSE 本社／ショールーム／通販部
〒321 栃木県宇都宮市竹林町503-1
TEL 0286-22-9811　FAX 0286-25-3970

Free Software Selection Vol.2情報/今回はパソコン通信をしていない方からもご応募いただけるよう考慮中です。詳細は次号で!



—グラデュエーション—





1月20日発売!

あの女子高生育成シミュレーション「卒業」が、いよいよX68000に登場! 多感で青春まっさかりの女生徒5人。彼女達を無事卒業までみちびいていくのが、教師であるあなたの使命。5人の生徒にかこまれた1年間を、あなたはどうすごしますか・・・?

■PC-9801版からの完全移植
美しいグラフィック、彼女達のセリフ…。すべてをPC-98版から忠実に移植しました。しかも画面デザインはX68000の機能に合わせてアレンジ。SEIKAのロゴが、画面をオシャレに彩ります。

■MIDI対応になりました!
X68000版はMIDI(GS音源)対応。だからBGMの迫力、美しさが違います。臨場感あふれるサウンドが、いっそうゲームを引き立てます。

●X68000版は画面のデザインが変更になります。

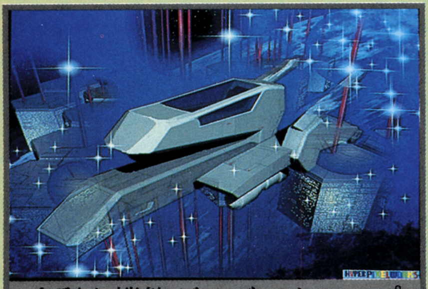
TAKERU価格 ¥7,800(税込)
■対応機種:X68000/X68030
■企画・制作:TAKERUソフト
Ⓒ 1992 JHV/HEADROOM

もはや、すべてが超越している

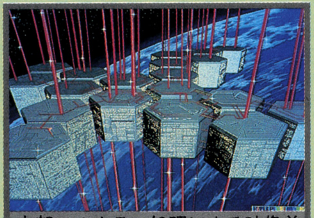
Industrial Magic
High Quality Design & Hyper Graphics Tools

Hyper Pixel Works

インダストリアルマジック　ハイパーピクセルワークス



多彩な機能が、バックアップ



内部フルカラー処理による映像美

ハイスピード・クイックレスポンスの快適なウィンドウシステム、アンチエリアスプリミティブレベル対応、内部フルカラー表現、高速・高性能ルーペシステム、多機能エフェクト、プリミティブ透明度処理、リアルタイムアンドゥ、リアルタイムターゲット、光学処理、プリミティブ合成処理、エクステンション機能、・・・CGは、新たな時代を迎えた。

TAKERU価格 ¥19,800(税込)
■X68000,030シリーズ/要2Mバイト
■企画・制作/ハイパー

好評発売中

■オプション 「エクステンション#1マルチフォントシステム」¥1,500 好評発売中!

2月号 1994年2月1日発行（毎月1回1日発行）通巻142号 昭和58年11月2日第三種郵便物認可 ソフトバンク株式会社発行 Printed in Japan 定価600円（本体583円）

T1002179020604 雑誌 02179-2